ioProgrammo

J2ME PUSH TECHNOLOGY
COME FARE IN MODO CHE IL TUO TELEFONINO REAGISCA AD EVENTI ESTERNI, AD ESEMPIO ALL'ARRIVO DI UN SMS

VERSIONE STANDARD RIVISTA+CD €6,90

PER ESPERTI E PRINCIPIANTI
Poste Italiane S.p.A Spedizione in A.P. • D.L. 353/2003 (conv.in L.27/02/2004 n.46) art.1 comma 2 DCB ROMA Periodicità mensile • FEBBRAIO 2006 • ANNO X, N.2 (99)

AL RIPARO DAGLI HACKER

Crea un software che controlla eventuali tentativi di attacco e te li segnala

TEORIA I principi base per svilupparlo con qualsiasi linguaggio

TECNICA sfruttiamo la Windows Management Instrumentation

PRATICA il codice con gli esempi in Visual Basic

VISUAL BASIC
CONFIGURARE WINDOWS VIA CODICE
Alla scoperta di WMI, il sistema che consente di "operare" al cuore del PC

XLS
TROVIAMO I DATI CHE CI INTERESSANO
XPATH, il linguaggio che consente di interrogare un file XML come se fosse un DB

QUESTION & ANSWER
PASSARE A VISUAL STUDIO 2005 CONVIENE?
Ti spieghiamo le caratteristiche del nuovo sistema. Quali gli inconvenienti?

PHP
FACCIAMO I PDF
Usa una classe che non richiede installazione, utilizzabile anche in hosting

CASI DI STUDIO: PHP
DATABASE ADDIO
Impara a creare applicazioni indipendenti dal db e rendi il tuo software disponibile per più aziende e sistemi

CASI DI STUDIO: JAVA
IMMAGINI PRONTE PER IL WEB
Realizziamo un software multipiattaforma che ridimensiona le immagini trovate nel disco rigido per poterle utilizzare sul WEB

SPECIALE
ioProgrammo BY EXAMPLE
22 ESEMPI COMPLETI
PER IMPARARE UN LINGUAGGIO IN MODO PRATICO E DIVERTENTE
VISUAL BASIC, VISUAL BASIC.NET, C#, PHP, JAVA, JAVASCRIPT, ASP.NET, HTML

VISUAL BASIC.NET
GESTIRE I FILE SENZA PROBLEMI
Tutto quello che devi sapere per salvare, creare, copiare e utilizzare i file nelle tue applicazioni

C#
MANOVRARE IL REGISTRY
Le funzioni base e le classi fondamentali per usare il registro di sistema

SFIDA TRA PROGRAMMATORI: CHI TROVERÀ IL MAGGIOR NUMERO DI CIFRE DECIMALI POSSIBILI DI PI GRECO?

Anno X - N.ro 2 (99) - Febbraio 2006 - Periodicità Mensile
Reg. Trib. di CS al n.ro 593 del 11 Febbraio 1997
Cod. ISSN 1128-594X
E-mail: ioprogrammo@edmaster.it
http://www.edmaster.it/ioprogrammo
http://www.ioprogrammo.it

Direttore Editoriale: Massimo Sesti
Direttore Responsabile: Massimo Sesti
Responsabile Editoriale: Gianmarco Bruni
Redazione: Fabio Farnesi
Collaboratori: M. Autiero, M. Bigatti, L. Buono, F. Grimaldi, F. Lippo, A. Lacava, M. Locuratolo, A. Pelleriti, R. Sidoti, F. Smelzo.

Segreteria di Redazione: Veronica Longo

Realizzazione grafica: Cromatika S.r.l.
Responsabile grafico: Paolo Cristiano
Coordinamento tecnico: Giancarlo Sicilia
Illustrazioni: M. Veltri
Impaginazione elettronica: Aurelio Monaco

Realizzazione Multimediale: SET S.r.l.
Realizzazione CD-Rom: Paolo Iacona

Pubblicità: Master Advertising s.r.l.
Via C. Correnti, 1 - 20123 Milano
Tel. 02 831212 - Fax 02 83121207
e-mail advertising@edmaster.it
Sales Director: Max Scortegagna
Segreteria Ufficio Vendite: Daisy Zonato

Editore: Edizioni Master S.p.a.
Sede di Milano: Via Ariberto, 24 - 20123 Milano
Tel. 02 831213 - Fax 02 83121330
Sede di Rende: C.da Lecco, zona industriale - 87036 Rende (CS)
Presidente e Amministratore Delegato: Massimo Sesti

ABBONAMENTO E ARRETRATI

ITALIA: Abbonamento Annuale: ioProgrammo (11 numeri) €59,90 sconto 20% sul prezzo di copertina di €75,90 · ioProgrammo con Libro (11 numeri) €75,90 sconto 30% sul prezzo di copertina di €108,90
Offerte valide fino al 31/03/06
Costo arretrati (a copia): il doppio del prezzo di copertina + € 5.32 spese (spedizione con corriere). Prima di inviare i pagamenti, verificare la disponibilità delle copie arretrate allo 02 831212.
La richiesta contenente i Vs. dati anagrafici e il nome della rivista, dovrà essere inviata via fax allo 02 83121206, oppure via posta a EDIZIONI MASTER via C. Correnti, 1 - 20123 Milano, dopo avere effettuato il pagamento, secondo le modalità di seguito elencate:

- cc/p n.16821878 o vaglia postale (inviando copia della ricevuta del versamento insieme alla richiesta);
- assegno bancario non trasferibile (da inviarsi in busta chiusa insieme alla richiesta);
- carta di credito, circuito VISA, CARTASI', MASTERCARD/EUROCARD, (inviando la Vs. autorizzazione, il numero della carta, la data di scadenza e la Vs. sottoscrizione insieme alla richiesta).
- bonifico bancario intestato a Edizioni Master S.p.A. c/o Banca Credem S.p.a. c/c 01 000 000 5000 ABI 03032 CAB 80880 CIN Q (inviando copia della distinta insieme alla richiesta).

SI PREGA DI UTILIZZARE IL MODULO RICHIESTA ABBONAMENTO POSTO NELLE PAGINE INTERNE DELLA RIVISTA. L'abbonamento verrà attivato sul primo numero utile, successivo alla data della richiesta.

Sostituzioni: qualora nei prodotti fossero rinvenuti difetti o imperfezioni che ne limitassero la fruizione da parte dell'utente, è prevista la sostituzione gratuita, previo invio del materiale difettato.
La sostituzione sarà effettuata se il problema sarà riscontrato e segnalato entro e non oltre 10 giorni dalla data effettiva di acquisto in edicola e nei punti vendita autorizzati, facendo fede il timbro postale di restituzione del materiale.
Inviare il CD-Rom difettoso in busta chiusa a:
Edizioni Master - Servizio Clienti - Via C. Correnti, 1 - 20123 Milano

Assistenza tecnica: ioprogrammo@edmaster.it

Servizio Abbonati:
☎ tel. 02 831212
@ e-mail: servizioabbonati@edmaster.it

Stampa: Arti Grafiche Boccia S.p.a. Via Tiberio Felice, 7 Salerno
Stampa CD-Rom: Neotek S.r.l. - C.da Imperatore - Bisignano (CS)
Distributore esclusivo per l'Italia: Parrini & C S.p.A.
Via Vitorchiano, 81 - Roma

Finito di stampare nel mese di Gennaio 2006

ioProgrammo by Example

Desieriamo ringraziarvi per la quantità di mail di ringraziamento che ci sono giunte per la rubrica ioProgrammo by Example. Ancora una volta avevamo visto giusto nel supporre che i programmatori avessero bisogno oltre che di un'ottima dose di applicazioni complete con la descrizione di nuove tecniche, anche di un buon supporto pratico che gli consentisse di risolvere rapidamente tutta una serie di problemi che spesso si verificano in fase di programmazione e che più che la logica coinvolgono la sintassi oppure le API, o i metodi da conoscere per realizzare quanto desiderato. Questo mese abbiamo focalizzato l'attenzione su Visual Studio 2005, lo abbiamo fatto perché il nuovo prodotto di Microsoft rappresenta una svolta epocale per quanto riguarda le politiche del gigante di Redmond. Prima di tutto perché almeno inizialmente sarà distribuito gratuitamente, in secondo luogo perché a livello concettuale il modo di programmare è cambiato radicalmente, il codice diventa sempre meno rispetto ai wizard e alle procedure visuali, tanto che imparare a usare Visual Studio è più un esercizio simile a imparare una qualunque applicazione che un linguaggio di programmazione, e questo non può essere che un bene sia per chi inizia e non deve fare i conti con la complessità dei moderni linguaggi, sia per chi vuole trarre la massima produttività dal proprio lavoro. Nonostante questo Visual Studio consente di programmare a molti livelli e gli esperti possono ancora fare riferimento al codice. Ci sembra una grande novità, forse la più grande in programmazione dall'epoca dell'avvento dei primi RAD. Certo Delphi e C++ Builder non sono da meno, e questo segna la strada che la programmazione sta seguendo. Non rimane che aspettare la risposta di PHP e Java, riusciranno questi due colossi a produrre ambienti produttivi quanto gli altri?

Fabio Farnesi
ffarnesi@edmaster.it

All'inizio di ogni articolo, troverete un simbolo che indicherà la presenza di codice e/o software allegato, che saranno presenti sia sul CD (nella posizione di sempre *\soft\codice* e *\soft\tools*) sia sul Web, all'indirizzo http://cdrom.ioprogrammo.it.

AL RIPARO DAGLI HACKER

Crea un software che controlla eventuali tentativi di attacco e te li segnala

- ✔ Teoria: I principi base per programmarlo con qualsiasi linguaggio
- ✔ Tecnica: quali funzioni di sistema usare e come usarle
- ✔ Pratica: il codice con gli esempi sviluppati in Visual Basic

pag. 14

DATABASE ADDIO

Impara come creare applicazioni indipendenti dal db e rendi il tuo software disponibile per più aziende e sistemi. Scopri la nuova classe PDO che rende semplice l'uso di più database pag. 22

SISTEMA

Visual Basic.NET e il File System pag. 66

Non c'è applicazione che non faccia accesso in qualche modo ai file, per salvare i dati, per gestire la configurazione, per scrivere l'output.Vediamo quali sono le classi per gestire al meglio file e directory

Le mani nel registro con .NET pag. 72
La libreria di base di .NET mette a disposizione gli strumenti necessari per accedere al registro di sistema, ma in alcuni casi è necessario ricorrere alle API di Windows

VISUAL BASIC

Configurare Windows via codice . . . . pag. 78
Con diversi esempi scopriamo i segreti di WMI – Windows Management Instrumentation – lo strumento che permette di controllare le caratteriste principali del sistema

MULTIMEDIA

Manipolare le immagini in Java . . . . pag. 84
Con il successo dei photoblog nasce il problema di ridimensionare le fotografie per metterle online. In questo articolo verrà realizzato un programma per visualizzare le immagini in una directory, selezionarle e ridimensionarle tutte con un clic

MOBILE

J2ME e la Push Technology . . pag. 92
Una delle caratteristiche più interessanti, introdotte con MIDP 2.0, è il cosiddetto Push Registry. Tramite esso, infatti, una MIDlet può essere attivata da un evento esterno, come un timer programmato

CORSI

XML • Usare XPATH per trovare i dati . . . . pag. 98
XSL è un linguaggio di trasformazione molto potente che ci consente di trasformare file XML in qualsiasi altro formato. Vediamo come selezionare solo i dati che ci interessano per trasformarli

SOLUZIONI

Questione Sfida di menti sul Pi Greco . . . . pag. 108
Da secoli gli scienziati si battono fra loro per trovare il maggior numero di cifre decimali possibili per il numero p. Vediamo quali sono gli algoritmi utilizzati dai moderni calcolatori

QUALCHE CONSIGLIO UTILE

I nostri articoli si sforzano di essere comprensibili a tutti coloro che ci seguono. Nel caso in cui abbiate difficoltà nel comprendere esattamente il senso di una spiegazione tecnica, è utile aprire il codice allegato all'articolo e seguire passo passo quanto viene spiegato tenendo d'occhio l'intero progetto. Spesso per questioni di spazio non possiamo inserire il codice nella sua interezza nel corpo dell'articolo. Ci limitiamo a inserire le parti necessarie alla stretta comprensione della tecnica.

RUBRICHE

IOPROGRAMMO by EXAMPLE

http://forum.ioprogrammo.it

BORLAND JBUILDER

FOUNDATION EDITION

RIVISTA + CD-ROM in edicola

L'AMBIENTE RAD PER PROGRAMMARE SUBITO IN JAVA E DISEGNARE FACILMENTE LE INTERFACCE!

Prodotti del mese

PostgreSQL 8.1

Il più professionale. Con la nuova gestione degli utenti che rende il database sempre più sicuro

Di PostgreSQL avevamo avuto modo di occuparci su ioProgrammo nei numeri precedenti con un bell'articolo approfondito. PostgreSQL è realmente un DB che dovrebbe occupare una posizione di maggiore rilievo nella scala dei database utilizzati a livello professionale. Non ha davvero nulla da invidiare a DataBase dal calibro di Oracle o di MSSql server e per velocità e numero di funzioni esposta si pone ad un livello molto più alto anche rispetto a MySQL 5.0 che comunque ha colmato parte del gap che aveva con l'ultima release.

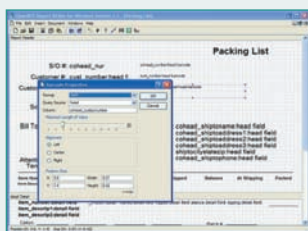
A fare la forza di MySQL è la completezza delle funzioni esposte che vanno dalle Stored Procedure, ai trigger, alla programmazione per mezzo di funzioni e alla modularità. Un grande database al quale manca forse soltanto una maggiore spinta di marketing per essere il leader del mercato.

[pag.105]

Open Report 1.1.1

Report per tutti i gusti, completo e gratuito, risolve il problema della generazione di Report in Java

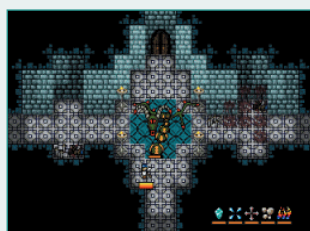
OpenReport è un sistema di reportistica completo e OpenSource. Il prodotto si presenta come molto efficace e dotato di tutte le funzioni tipiche di un tool per il report dei dati. Uno dei suoi maggiori punti di forza consiste nel poter salvare la definizione del report direttamente in formato XML e questo lo rende particolarmente adatto ad interfacciarsi con un gran numero di applicazioni. D'altra parte il suo orientamento è quello di lavorare in congiunzione con PostgreSQL anche se non dovrebbe essere complesso modificarlo per renderlo compatibile con altri tipi di db. Il suo codice è opensource e modulare, quindi se trovate che i suoi meccanismi interni siano di vostro gradimento potete sempre modificarlo per aggiungere le parti che possono aiutarvi nel vostro lavoro.

[pag.106]

FreeBasic 15b

Un Basic con licenza Gnu. Per i nostalgici del passato ma anche per sviluppare utility a riga di comando

Questo compilatore farà la felicità di un gran numero di affezionati, che si troveranno a disposizione un completo compilatore Basic a linea di comando esattamente come ai bei tempi dellinizio dell'informatica.

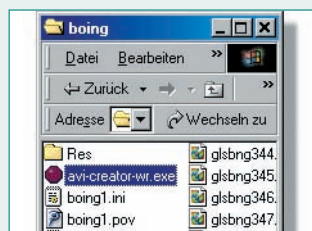
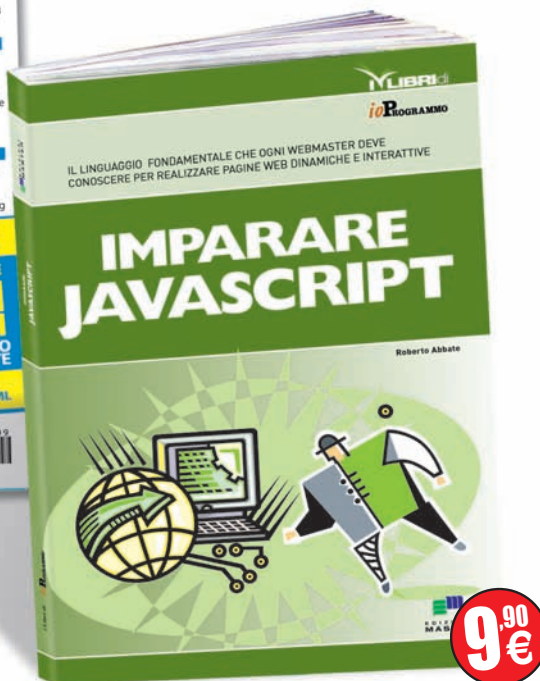
Al di la dellaspetto nostalgico bisogna però sottolineare che FreeBasic risulta utilissimo nel confezionare piccole ma potenti utility in modo rapido che possono automatizzare il comportamento del sistema. Il compilatore viene distribuito con una serie impressionante di librerie che vanno dalle GD per la creazione di interfacce grafiche alle fastcgi per la creazione di piccoli script adatti al web. Si tratta di un'opportunità importante che consente al contempo di lavorare con un linguaggio semplice e potente.

[pag.107]

Avi Creator 1.5

Per creare piccole animazioni da inserire nei tuoi programmi

Avi Creator appartiene a quella schiera di utility che non si possono definire esattamente dedicate ai programmatori ma che aiutano senza dubbio a creare interfacce esteticamente accattivanti e che restituiscono un maggiore feedback allutente. Si tratta di un software che riesce a creare delle minianimazioni in formato AVI assemblandole da una lista di bitmap. Utili per sottolineare ad esempio operazioni come la copia dei file.

L'uso del programma é piuttosto semplice e gli effetti sono interessanti. In questo modo si possono ottenere applicazioni che restituiscono un feedback anche visivo all'utente che non é costretto a rimanere per lungo tempo in attesa della conclusione di operazioni che non vengono in alcun modo segnalate

[pag.107]

RIVISTA + LIBRO + CD-ROM in edicola

I contenuti del libro

Imparare JAVASCRIPT

Scrivere pagine Web Dinamiche, è questa l'aspirazione di ogni Web Master che si rispetti. L'idea è di condurre l'utente per mano attraverso la navigazione, distraendolo dalla complessità dell'interfaccia per condurlo nella fruizione dei contenuti. Nonostante l'evoluzione del Web, il linguaggio principe per l'interazione pagine web/utente rimane JavaScript. Si tratta di uno dei pochi metodi che il WebMaster ha a disposizione per animare bottoni, textbox, menu e ogni tipo di elemento posizionabile su una pagina HTML. Roberto Abbate, con uno stile semplice, chiaro, lineare, mai dispersivo, rimane concentrato sull'obiettivo e porta il lettore a imparare il linguaggio tramite una miscela di esempi e commenti dettagliati che rendono il percorso di apprendimento semplice ed immediato.

Il linguaggio fondamentale che ogni webmaster deve conoscere per realizzare pagine web dinamiche e interattive

- La sintassi di base del Linguaggio
- L'interazione con l'HTML
- Il modello ad Oggetti
- Browser, Document e Windows oggetti fondamentali

News

RILASCIATA L'ECLIPSE WEB PLATFORM 1.0

Come noto Eclipse è la piattaforma universale ed estendibile dedicata agli sviluppatori. La versione 1.0' Web Platform contiene una serie di strumenti che consentono uno sviluppo facilitato di applicazione Web in ambiente J2EE.
L'intento è quello di fornire ai vari vendors una piattaforma utilizzabile anche in ambiente di business, la dove le applicazioni enterprise basate su un modello distribuito costituiscono l'unico mercato possibile per aziende ormai evolute, che hanno recepito che la modularizzazione del software rappresenta l'unica via possibile per uno sviluppo efficace.
Eclipse si è già distinta negli ultimi tempi per avere prodotto una delle piattaforme di sviluppo più innovative che hanno modificato radicalmente le abitudini dei programmatori, tanto si è creata una comunità piuttosto vasta che usa Eclipse come mezzo principe per lo sviluppo in Java

STRETTA DI MANO FRA ORACLE E IBM

Le due società storiche rivali in ambiente database hanno sotterrato l'ascia di guerra per dedicarsi a un progetto comune che promuova l'integrazione di software Oracle all'interno della piattaforma AIX di IBM.
L'operazione non è da poco se si pensa che insieme i due colossi detengono una larga parte del mercato Enterprise, e che il sistema operativo AIX è quasi uno standard per organizzazione di grandi dimensioni. D'altra parte Oracle è un gigante nel mercato dei database. In questo modo Larry Ellison, CEO di Oracle intende dare un colpo al cerchio e uno alla botte, da un lato accarezzando Sun e il suo Solaris, dall'altro stringendo la mano a IBM e AIX in un'operazione che fa di Oracle il centro di un mercato remunerativo ed altamente in espansione

LA TOP 10 DI MICROSOFT

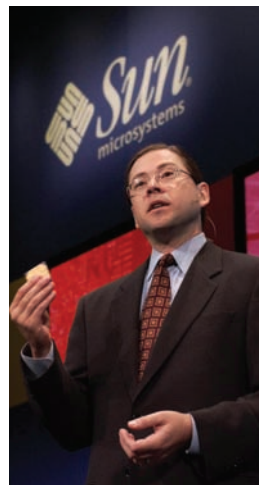
Anche quest'anno la "Direction On Microsoft", ovvero l'organizzazione "indipendente" che si occupa di monitorare le strategie del colosso di Redmond ha tracciato le linee guida di quelli che dovrebbero essere i cambiamenti strategici che determineranno il cammino di Microsoft nel 2006. L'intera Top Ten è disponibile all'indirizzo http://www.directionsonmicrosoft.com/2006top10.html. La traccia più significativa sembra essere dettata dall'arrivo di Windows Vista, che dovrebbe migliorare di molto l'apporto alla produttività e alla sicurezza delle piccole e medie imprese. Tuttavia si registra anche qualche preoccupazione in merito al gran numero di effetti grafici e animazioni che caratterizzano il nuovo sistema operativo e che in qualche caso potrebbero disorientare l'utente medio. Il secondo punto su cui ruotano le strategie di

PIRATERIA: È GELO FRA USA E RUSSIA

L'America ha tuonato: la Russia potrà entrare nell'organizzazione mondiale per il commercio solo se attuerà una politica forte e determinata contro la violazione dei diritti sulla proprietà intellettuale. L'avvertimento del Senato americano suona come un monito forte e chiaro alla salvaguardia degli scambi commerciali fra i due paesi, ma è anche il simbolo di quanto la moderna società attraversi un periodo di transizione abbastanza complesso, la dove la difficoltà di controllo di un così vasto mercato rappresenta un ostacolo difficilmente superabile. D'altra parte la Russia rappresenta il bacino di concentrazione di una gran parte di pirati informatici che proprio in virtù, anche, della scarsa volontà politica di arginare il problema trovano terreno fertile per le proprie attività. Allo stato attuale i politici russi non sembrano avere risposto, o quantomeno non sembrano averlo fatto in modo adeguato a quelle che sono le richieste effettuate dalla superpotenza Americana. D'altra parte gli interessi in gioco sono diversi e l'approccio americano ai diritti sul Software è decisamente differente rispetto a quello russo. Difficilmente il problema potrà essere risolto, nonostante i moniti dell'America e anche se la Russia dovesse piegarsi alle richieste degli USA. Di fatto la sola guerra alla pirateria non può rappresentare una soluzione se non accompagnata da una mediazione e da una discussione forte sull'interpretazione dei diritti d'autore. Discussione che è in corso ormai da parecchi anni e che rappresenta il perno importante su cui sta girando buona parte della società informatica moderna

Microsoft dovrebbe concernere affidabilità e sicurezza. Questa non è una grande novità, da anni la società di Bill Gates combatte la propria personale guerra contro i bug di sicurezza, e da anni è tacciata di produrre sistemi poco affidabili. A onor del vero non possiamo dire che non ci siano stati miglioramenti da questo punto di vista. Windows nel corso delle sue varie versioni è diventato più stabile e in parte più sicuro, anche se è praticamente impossibile produrre un sistema così largamente distribuito senza incappare in qualche problema di sicurezza e affidabilità. Per quello che ci riguarda spicca il "Get Going On Tools" dove si proclama la volontà di MS di supportare i programmatori con strumenti di sviluppo a basso costo, questo dovrebbe contestualmente aumentare il numero di tools disponibili per il sistema operativo. Se questa "promessa" fosse mantenuta, non potremmo che esserne contenti!

SUN APRE A LINUX

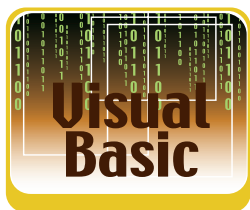
La vocazione di Sun all'OpenSource si rivela sempre più prepotentemente. È recente la notizia secondo cui il codice di Janus è stato reso disponibile alla community degli sviluppatori. Janus è il framework che consente ad applicazioni Linux di girare su piattaforma OpenSolaris. Per l'occasione il progetto in questione è stato ribattezzato BrandZ. Da un lato questa mossa di Sun è stata apprezzata dal vasto pubblico aderente agli schemi dell'OpenSource, dall'altro questa svolta imperiosa di Sun non arriva inaspettata e l'abbraccio al modello del software libero sta portando sul mercato una nutrita serie di applicazioni proprio basate su Java. Nel mese di Dicembre per la prima volta il numero di applicazioni Java ospitate su SourceForge ha superato quello di progetti scritti in C++ a testimonianza dell'efficacia che questo genere di modello apporta nel numero e nella qualità delle applicazioni sviluppate. Bisogna anche dire che se il software sviluppato in ambiente Linux è in grado di girarre su OpenSolaris questo non può essere che un vantaggio per il sistema operativo di Sun.

SEAGATE COMPRA MAXTOR

La notizia è di quelle che scotta. Seagate, leader indiscusso del mercato dello storage del quale detiene circa il 30% ha appena lanciato la propria proposta d'acquisto a Maxtor che ne detiene circa il 17%. La fusione delle due aziende porterebbe alla nascita di un colosso con il controllo del 47% dell'intero mercato mondiale degli Hard Disk. Sarebbe Seagate ad avere i maggiori vantaggi da quella che più che una fusione sembra un acquisizione in piena regola. La nuova società manterrebbe, infatti il nome di Seagate e erediterebbe l'intero team dirigenziale da quest'ultima. L'intera operazione potrebbe essere finalizzata nel corso della seconda metà del 2006. I risvolti per il mercato sarebbero notevoli. Di fatto un gigante di queste porzioni sarebbe in grado di apportare un notevole sviluppo al settore dello storage e di proporre prezzi sempre più competitivi. D'altra parte la storia ci ha insegnato che l'assenza di concorrenza non è un bene, bisogna capire se la diretta concorrente "Western Digital" che detiene attualmente il 17% del mercato sarà in grado di reggere l'impatto di questa nuova acquisizione

ARRIVA LA TERZA GENERAZIONE DI NESSUS

Per chi non lo sapesse Nessus è uno dei Vulnerability Scanner più diffusi della rete. Sostanzialmente analizza un sistema alla caccia dei possibili buchi di sicurezza. La versione 3 rappresenta prima di ogni cosa un punto di discontinuità con le versioni precedenti della licenza, non sarà possibile infatti distribuire i sorgenti. In secondo luogo il tool è stato ampliato con un maggior numero di strumenti e la gestione dei plugin è stata migliorata. Il punto di forza di Nessus è sempre stata infatti la sua modularità, e la disponibilità di numerosi plugin che ne estendono le funzionalità. Il linguaggio con cui questi plugin vengono scritti è il NASL, un linguaggio di Scripting abbastanza comodo ed efficiente, ed è proprio il passaggio all'engine NASL3 che rende questa nuova versione di Nessus totalmente diversa da quelle passate, di fatto il passaggio al nuovo engine ha comportato il dover riscrivere quasi totalmente il core del tool. Nessus funziona su un modello Client/Server la dove la macchina da analizzare deve avere installato il demone per il suo controllo, a significare che non si tratta di un tool di Hacking ma di un vero e proprio tool per l'analisi e la sicurezza dei sistemi. Nella prima metà del 2006 si attende la versione per Windows. Al momento il software è disponibile in ambiente Linux

A caccia di intrusi

In questo articolo realizzeremo un programma in grado di capire se qualcuno sta attaccando il nostro sistema e che ci avvertirà in tempo utile nel caso in cui dovesse rilevare "anomalie"

Un IDS può essere visto come un sistema hardware e/o software che consente ad un amministratore di sistema di accorgersi di qualunque situazione "strana" che possa indicare un tentativo d'intrusione portato o meno a buon fine. Da quanto appena affermato si deduce facilmente che un IDS deve essere costituito da un insieme di strumenti in grado di rilevare tutti i pacchetti sospetti che transitano sulla nostra rete. Si deve subito intuire che un IDS non è comunque un firewall, nel senso che solitamente non si occupa di bloccare qualcosa, ma permette altresì di tenere sotto controllo una certa situazione ed avvertirci, in qualche modo, su quanto di strano sta succedendo.
Una prima classificazione dei sistemi di intrusion detection può essere fatta in base alla metodologia utilizzata per rilevare il tentativo d'intrusione:

- IDS basati sulle firme (anche conosciute come signature);
- IDS basati sulle regole.

I sistemi IDS basati sulle firme, com'è facile intuire, basano la loro strategia di controllo sulla definizione di pattern che consentono di decidere se un sistema è sotto la minaccia d'intrusi oppure no. Un analogo sistema simile a questo è rappresentato dal comune antivirus. I sistemi IDS basati sulle regole, invece, rilevano i tentativi d'intrusione attraverso la definizione di regole che consentono di stabilire ciò che deve essere considerato normale e ciò che va tenuto sotto controllo perché sospetto. Una seconda classificazione di questi sistemi riguarda il loro "raggio d'azione".
A questo proposito distinguiamo:

- **IDS host-based (HIDS):** questo genere di IDS richiedono la loro implementazione su ogni host che si vuole proteggere. Ovviamente possono essere considerati i più efficaci se visti nell'ottica d'implementare i controlli di sicurezza indipendentemente dall'ambiente nel quale ci si trova. Un sistema IDS di tipo host-based analizza diverse aree per determinare attività illecite o "semplicemente" la presenza di un intruso. Gli IDS host-based, possono servirsi di diversi tipi di file log che consultano per confrontare le registrazioni con un database interno di firme che rappresentano gli attacchi già conosciuti.

- **IDS network-based (NDIS):** gli IDS del tipo *network-based* si differenziano dai precedenti perché inviano i pacchetti attraverso un dispositivo singolo "prima" di inviarli a host specifici. Per molti aspetti sono considerati i meno sicuri perché, considerata l'enorme crescere di dispositivi portatili, non è possibile implementare tali controlli in maniera efficace. Molti IDS *network-based*, richiedono che il dispositivo di rete del sistema host sia impostato sulla modalità promiscuous. Questa modalità consente di catturare ogni pacchetto inviato sulla rete.

- **IDS ibridi:** sono sistemi che implementano, all'interno di un'architettura di rete, sistemi HIDS e NIDS.

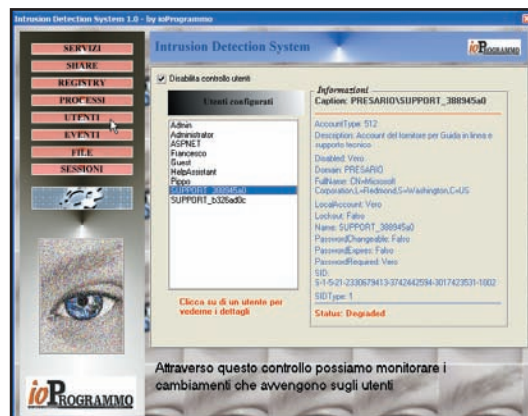
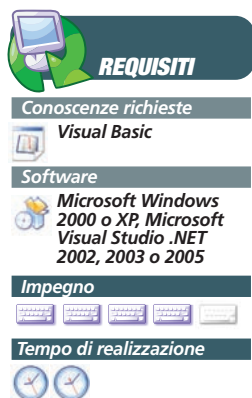



Fig. 1 I vari servizi che si possono controllare

Esistono diversi vantaggi e svantaggi sull'utilizzo di un HIDS o di un NDIS e la scelta dipende da diversi fattori economici e infrastrutturali. Per gli scopi di questo articolo ci soffermeremo principalmente sui sistemi HIDS ed implementeremo un mini IDS in Visual Basic sfruttando, peraltro, le capacità offerte da WMI.

IL PROGETTO IN VB

Il progetto in Visual Basic è costituito da alcuni form e da pochi moduli che racchiudono tutte le funzioni e le procedure utilizzate dal programma. In particolare:

- **frmPrincipale:** rappresenta la form principale dalla quale poter fare le proprie scelte;
- **frmAllarmi:** mostra una scheda che visualizza i controlli correntemente attivi;
- **frEventi:** consente l'inserimento di nuovi *Event ID* da controllare attraverso il *Security Log;*
- **CaricaStatus.bas:** raccoglie una serie di procedure che consentono di avere lo stato attuale sui servizi, processi, utenti, ecc;
- **Generale.bas:** contiene, in particolare, la funzione *ImpostaWQLQuery()* per il caricamento dell'array relativo alle query WQL;
- **Registry.bas:** al suo interno sono presenti diverse funzioni per agire sul *Registry*. Alcune non sono sono utilizzate, ma potrebbero tornare utili;
- **Settings.bas:** contiene le procedure che permettono la creazione della struttura del *Registry* (dal punto di vista delle chiavi utili al programma), la sua lettura ed il salvataggio delle impostazioni.

Il programma realizzato, attraverso l'utilizzo di WMI, consente di controllare:

- Modifiche sullo stato dei servizi;
- Creazione/eliminazione di condivisioni sulla macchina locale;
- Modifiche al *Registry* (in corrispondenza delle chiavi di tipo *Run* e *RunOnce* dell'hive *HKEY_ LOCAL_MACHINE*) per rilevare eventuali tentativi di avviare programmi in automatico;
- Creazione/eliminazione di processi pericolosi;
- Modifiche sugli utenti;
- Registrazione, nel *Security Log*, di tentativi d'intrusione. Il controllo, come vedremo, si basa sul monitoraggio di un gruppo di eventi aventi un determinato ID;
- Inserimento o eliminazione di job schedulati;
- Creazione/distruzione di sessioni remote verso il proprio PC.

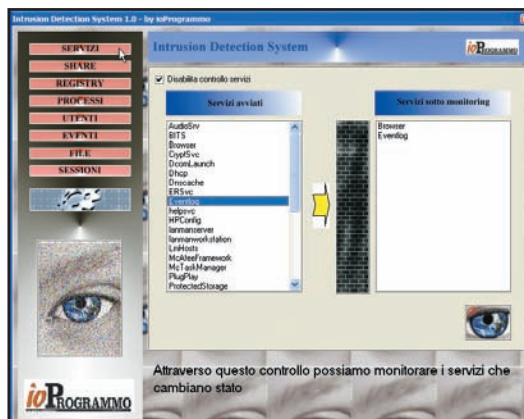



Fig. 2: **L'interfaccia principale del programma**

Le tecniche utilizzate nel programma sono molto simili, almeno in parte, a quelle implementate nel progetto pubblicato qualche numero fa su ioProgrammo relativo al monitoring in VB attraverso SNMP e WMI. Pertanto, rimandiamo ai quei numeri eventuali spiegazioni sulle chiamate a funzioni e procedure che utilizzano oggetti "prettamente WMI", soffermando maggiormente la nostra attenzione sulle parti del programma "particolari".

FUNZIONAMENTO DEL PROGRAMMA

Il programma realizzato implementa un mini IDS basandosi su alcuni controlli standard che potrebbero rilevare, se effettuati in tempo reale o a brevi intervalli di tempo, un possibile tentativo d'intrusione. All'avvio, il form principale si preoccupa di svolgere alcune azioni importanti:

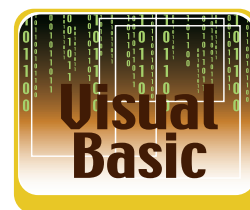
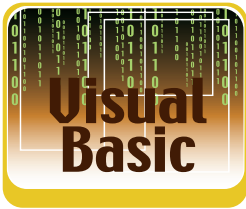
- Riempire alcune listbox con i dati attuali relativi a servizi, condivisioni, utenti, ecc…
- Ridimensionare e posizionare i pannelli eventualmente riposti in maniera disordinata in *design-time;*
- Controllare se si è alla prima installazione e creare eventualmente la struttura del *Registry* utile al programma;
- Valorizzare correttamente l'array *WQLQuery* (richiamando l'apposita procedura *ImpostaWQLQuery*);
- Avviare, attraverso il controllo di appositi flag, le procedure di monitoraggio attivate dall'utente.

Come si può facilmente intuire dall'interfaccia principale, la form *frmPrincipale,* è suddivisa in due parti. La parte sinistra raccoglie una serie di pulsanti che consentono l'abilitazione e la configurazione dei vari controlli. Nella parte destra, ovviamente, troviamo i dettagli di ciascun controllo e i possibili parametri interessati.
Partiamo quindi dalle procedure che caricano le listbox presenti sulla form *frmPrincipale* per passare poi alle procedure di controllo vere e proprie.

Fig. 3: ***Un esempio di caricamento dei processi***

IL CARICAMENTO DELLE INFORMAZIONI

Il programma si preoccupa, all'avvio, di riempire una serie di controlli listbox con le informazioni del sistema locale. Per far questo sfrutta le classi WMI apposite e la struttura di ogni singola procedura è pressappoco la stessa. Per queste ragioni, mostreremo di seguito la sola procedura *ListaServizi()* che si occupa di caricare tutti i servizi attualmente avviati.

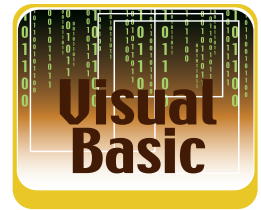
```
Public Sub ListaServizi(Optional NameService)

    Dim objWmiLocator As New
                        WbemScripting.SWbemLocator
    Dim objWmiServices As SWbemServices

    Set objWmiServices=
        objWmiLocator.ConnectServer(".","root\CIMV2")
    objWmiServices.Security_.ImpersonationLevel=3
    objWmiServices.Security_.Privileges.AddAsString
                                ("SeSecurityPrivilege")

    'Carica la lista dei servizi correnti
      Set colitems=objWmiServices.ExecQuery("SELECT
      * FROM Win32_Service WHERE StartMode='Auto'
                        AND State='Running'", "WQL",
                        wbemFlagReturnImmediately+
                                bemFlagForwardOnly)

    'Carica lstProcess con la lista dei servizi correnti
      For Each Servizio In colitems
          frmPrincipale.lstServices.AddItem Servizio.Name
      Next

End Sub
```

Questa procedura non fa altro che eseguire una query, scritta in linguaggio pseudo-SQL *(WQL),* utilizzando la classe *Win32_Service,* alla ricerca di tutti i servizi in modalità *Automatic* e attualmente in stato *Running.*

Nel caso dei processi, degli utenti, delle condivisioni, ecc. le classi utilizzate sono diverse, ma la tecnica di recupero delle informazioni è la medesima. Al termine di questa procedura, la form principale è mostrata a video con tutti i dati trovati. Ogni pannello, come si sarà potuto notare, consente di abilitare /disabilitare ciascun controllo. Il generico click del mouse su qualunque checkbox che controlla lo stato di un determinato allarme, provoca un paio di azioni possibili ossia: se il flag è impostato a 1 (monitoring disabilitato), viene distrutto l'oggetto di tipo *WBemSink* associato a quel controllo; nel caso contrario, invece, lancia la procedura che consentirà il monitoring. Riferendoci sempre al caso dei servizi, la procedura che si preoccupa di abilitare il controllo, è la *ServiceStatus():*

```
Public Sub ServiceStatus()

    Dim objWmiLocator As New
                        WbemScripting.SWbemLocator
    Dim objWmiServices As SWbemServices

    ' Connessione al namespace della macchina locale
      Set ServiceSink=New SWbemSink
      Set objWmiServices=
          objWmiLocator.ConnectServer(".", "root\CIMV2")
      objWmiServices.Security_.ImpersonationLevel=3
      objWmiServices.Security_.Privileges.AddAsString
                                ("SeSecurityPrivilege")

    ' Controlla eventuali servizi down
      objWmiServices.ExecNotificationQueryAsync
                            ServiceSink, WQLQuery(5)

End Sub
```

mentre la procedura Visual Basic che "cattura" l'evento è:

```
Public Sub ServiceSink_OnObjectReady(ByVal
```

```
StatusEvent As SWbemObject, ByVal EventContext
                    As SWbemNamedValueSet)

  ' Controlla che il servizio sia monitorato
    If ChkListDuplicates(lstServiceAlarm.hWnd,
              StatusEvent.TargetInstance.Name) Then
        MsgBox "Il servizio " &
                    StatusEvent.TargetInstance.Name &
                  " ha cambiato stato. Attualmente è " &
                       StatusEvent.TargetInstance.State
        DoEvents
    End If

End Sub
```

Pria di passare oltre, è bene osservare un particolare presente peraltro in molte delle procedure di tipo *_OnObjectReady* e simili ossia il richiamo alla funzione *ChkListDuplicates()*. Attraverso questa funzione e, soprattutto, grazie all'uso della *SendMessage()* possiamo popolare una seconda lista (un controllo listbox) con l'elenco dei servizi e dei processi da controllare (senza possibilità di ripetizioni), semplicemente cliccando sulla lista principale tramite il mouse.
Effettuando l'operazione inversa, invece, elimineremo dalla seconda lista i processi ed i servizi precedentemente controllati:

```
Function ChkListDuplicates(chwnd As Long,StrText
                               As String) As Boolean
    ChkListDuplicates=(SendMessageByString(
        chwnd,LB_FINDSTRINGEXACT,-1,StrText)>-1)
End Function
```

In questa maniera, quando scatta l'evento associato al controllo che stiamo considerando, se "l'oggetto" si trova nella lista di quelli da controllare, viene avvertito l'utente.

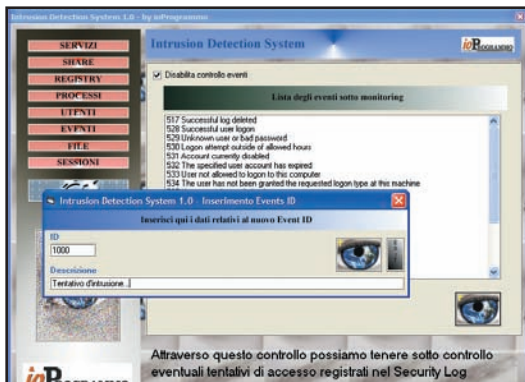



Fig. 4: ***L'inserimento di nuovi Event ID la monitorare***

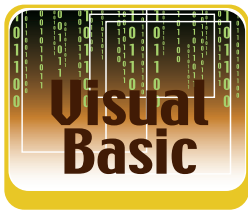
Un procedimento simile è stato seguito anche per il controllo dei tentativi di accesso registrati nel *Security Log*. In questo caso, tuttavia, è stata implementata anche la possibilità d'inserimento di nuovi *Event ID* non presenti nella lista di default.

IL CONTROLLO DEL REGISTRY

Uno dei modi per arrecare danni ad un sistema, dopo essersi introdotti, è quello di prevedere l'aggiunta di particolari programmi all'avvio del PC, in modalità silent. In particolare, quest'azione può essere svolta senza problemi modificando principalmente le seguenti chiavi di registro:

```
Windows NT 4.0, Windows 2000, Windows XP and
                              Windows Server 2003:
"   HKEY_LOCAL_MACHINE\Software
            \Microsoft\Windows\CurrentVersion\Run
"   HKEY_CURRENT_USER\Software
            \Microsoft\Windows\CurrentVersion\Run
" HKEY_LOCAL_MACHINE\Software
        \Microsoft\Windows\CurrentVersion\RunOnce
"  HKEY_CURRENT_USER\Software
        \Microsoft\Windows\CurrentVersion\RunOnce
Windows 95/98/ME:
"   HKEY_LOCAL_MACHINE\Software
            \Microsoft\Windows\CurrentVersion\Run
"   HKEY_CURRENT_USER\Software
            \Microsoft\Windows\CurrentVersion\Run
"   HKEY_LOCAL_MACHINE\Software
        \Microsoft\Windows\CurrentVersion\RunOnce
"   HKEY_CURRENT_USER\Software
        \Microsoft\Windows\CurrentVersion\RunOnce
"   HKEY_LOCAL_MACHINE\Software
     \Microsoft\Windows\CurrentVersion\RunServices
"   HKEY_LOCAL_MACHINE\Software\Microsoft
          \Windows\CurrentVersion\RunServicesOnce
"   HKEY_LOCAL_MACHINE\Software\Microsoft
            \Windows\CurrentVersion\RunOnce\Setup
```

Considerando di lavorare su sistemi presenti nel primo gruppo, il programma si preoccupa di controllare eventuali modifiche di queste ed altre chiavi. In particolare, per una limitazione delle funzioni che vedremo, il solo hive controllato è *HKEY_LOCAL_ MACHINE*.
Di seguito riportiamo la generica procedura che avvia il monitoring:

```
Public Sub RegistryStatus()

   Dim objWmiLocator As New
                  WbemScripting.SWbemLocator
   Dim objWmiServices As SWbemServices

   'Connessione al namespace della macchina locale
```

```
    Set RegistrySink=New SWbemSink
    Set objWmiServices=
      objWmiLocator.ConnectServer(".","root\default")
    objWmiServices.Security_.ImpersonationLevel=3
    objWmiServices.Security_.Privileges.AddAsString
                                   ("SeSecurityPrivilege")

  'Controlla eventuali modifiche al Registry
    objWmiServices.ExecNotificationQueryAsync
                               RegistrySink,WQLQuery(9)

End Sub
```

Rispetto alle "solite" procedure di questo genere, si osservino almeno due cose. La prima è che il namespace utilizzato non è più *root\CIMV2*, ma *root\default*.

Fig. 5: ***Lista dei controlli implementati ed attivi***

La seconda è relativa alla query utilizzata:

```
WQLQuery(9)="SELECT * FROM
                RegistryKeyChangeEvent WHERE Hive=
                'HKEY_LOCAL_MACHINE' AND (KeyPath=
        'Software\\Microsoft\\Windows\\CurrentVersion
  \\Run' OR KeyPath='Software\\Microsoft \\Windows
           \\CurrentVersion\\RunOnce' OR KeyPath=
        'Software\\Microsoft\\Windows\\CurrentVersion
                      \\RunOnceEx' OR KeyPath=
                      'Software\\Microsoft\\Windows
                 \\CurrentVersion\\RunServices' OR
              KeyPath= 'Software\\Microsoft\\Windows
              \\CurrentVersion\\RunServicesOnce'
                    OR KeyPath='Software\\Microsoft
        \\Windows\\CurrentVersion\\RunServicesEx')"
```

che sfrutta una classe apposita per questi eventi denominata *RegistryKeyChangeEvent*. Per maggiori dettagli su questa e altre classi che si occupano di modifiche sul *Registry*, consigliamo di avvalersi del link Microsoft *http://msdn.microsoft.com/library/default .asp?url=/library/en-us/wmisdk/wmi/wmi_registry_classes.asp*.

CONTROLLO DEI TASK SCHEDULATI

Il controllo dei task schedulati è un'altra importante funzione che dovrebbe essere sempre svolta da qualunque programma d'*intrusion detection*. La procedura che si occupa di svolgere questa funzione è la *ControllaFile()*. Il nome della procedura, in realtà, è forviante per gli scopi che ci siamo prefissi, ma fra pochissimo spiegheremo il perché sia stata dichiarata in questo modo.
Innanzitutto eccola:

```
Public Sub ControlloFile()

  Dim objWmiLocator As New
                        WbemScripting.SWbemLocator
  Dim objWmiServices As SWbemServices

  'Connessione al namespace della macchina locale
    Set objWmiServices=
      objWmiLocator.ConnectServer(".","root\CIMV2")
    objWmiServices.Security_.ImpersonationLevel=3
  'Controlla le sessioni che vengono create
    Set FileCreateSink=New SWbemSink
    objWmiServices.ExecNotificationQueryAsync
                           FileCreateSink,WQLQuery(6)
  'Controlla le sessioni che vengono eliminate
    Set FileDeleteSink=New SWbemSink
    objWmiServices.ExecNotificationQueryAsync
                           FileDeleteSink,WQLQuery(7)
End Sub
```

La generica query che viene richiamata è la seguente:

```
WQLQuery(6)="SELECT * FROM
          __InstanceCreationEvent WITHIN 1 WHERE
 Targetinstance ISA 'CIM_DirectoryContainsFile' AND
 TargetInstance.GroupComponent='Win32_Directory
        .Name=""" & PathScheduledTask & """" & ""
```

dove *PathScheduledTask* memorizza il path sul proprio disco rigido dove ritroviamo i job schedulati (solitamente pari a *C:\WINDOWS \TASKS*).La particolarità di questa query è nell'utilizzo della *CIM_DirectoryContainsFile* che ci permette, opportunamente utilizzata (come nel nostro caso), di tener traccia di ogni cambiamento avvenga all'interno di una determinata cartella. Ogni qualvolta, quindi, viene creato o eliminato un job, l'evento associato avverte l'utente di queste modifiche. La classe *CIM_DirectoryContainsFile* è molto interessante dal punto di vista del monitoraggio perché consente di specificare le cartelle "sensibili" che occorre controllare. Tuttavia, qualora non fosse stretta-

mente necessario tener traccia di "ogni" cambiamento, ma solo di qualche file, può essere utilizzata una classe più appropriata (dal punto di vista dell'occupazione delle risorse anche moto più "leggera") denominata *CIM_DataFile*. Ecco di seguito la generica query che consente il controllo su eventuali modifiche apportate al *Boot .ini:*

```
"SELECT * FROM __InstanceOperationEvent within 10
        WHERE TargetInstance ISA 'CIM_DataFile' AND
                TargetInstance.Name ='c:\\Boot.ini'"
```

LA LISTA DEI TASK

L'insieme delle procedure che si occupano di gestire la creazione e l'eliminazione dei task schedulati comprendono ne comprendono anche una denominata *ListaJobSchedulati()*. Attraverso essa, otteniamo la lista di tutti i job attualmente schedulati sulla macchina.

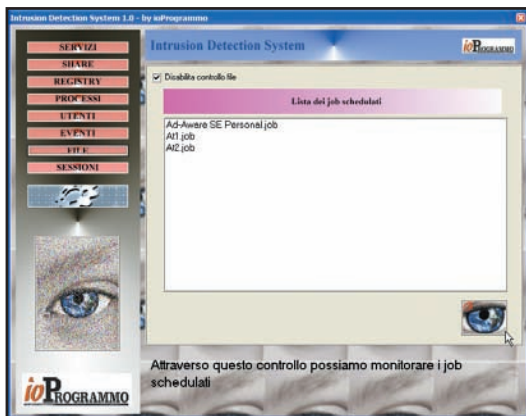



Fig. 6: **Lista dei job schedulati**

A questo proposito, prima di vedere in dettaglio l'implementazione di questa procedura, è bene osservare quanto segue. La lista che vogliamo ottenere poteva essere ottenuta "apparentemente" anche attraverso l'apposita classe *WMI Win32_ScheduledJob*. Tuttavia, facendo alcuni test, si evince subito che questa classe non consente di ottenere la lista dei job creati attraverso la procedura guidata Operazioni pianificate, bensì soltanto quegli item definiti attraverso l'uso del comando *AT*. Per ovviare a questo inconveniente (che, peraltro, ha determinato la scelta sull'opportuno codice da implementare per il controllo di questi eventi), si è fatto uso del *File System Object*, ottenendo la lista di tali file contenuti all'interno della cartella *%windir%\Tasks*. Ecco la procedura *ListaJobSchedulati():*

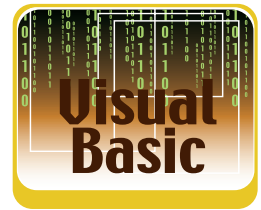
```
Public Sub ListaJobSchedulati()
Dim PathJobSchedulati As String
   'Preleva tutti i files nella directory %WINDIR%\Tasks
     PathJobSchedulati=Environ("windir")&"\Tasks"

     Set Fso=CreateObject(
                          "Scripting.FileSystemObject")
     Set FolderObject=Fso.GetFolder(
                                   PathJobSchedulati)
     Set FilesObject=FolderObject.Files

'Dalla collection files controlla se ci sono file .job

  For Each file In FilesObject
      'Se ci sono file .job
       If Fso.GetExtensionName(file)="job" Then
           frmPrincipale.lstScheduledJob.AddItem
                                            file.Name
        End If
  Next

'Elimina gli oggetti appena creati dalla memoria

  Set FileObject=Nothing
  Set FolderObject=Nothing
  Set Fso=Nothing

End Sub
```

Attraverso l'utilizzo del *File System Object*, possiamo controllare se esistono o meno file con estensione *.Job* all'interno della cartella *Tasks* e caricare di conseguenza la il controllo *lstScheduledJob* presente sulla form principale.

CONCLUSIONI

Lo sviluppo di un sistema d'*intrusion detection* è, senza dubbio alcuno, una delle operazioni più complicate che possono coinvolgere un programmatore. La difficoltà principale sta soprattutto nella conoscenza dei punti deboli di un sistema e, ancor di più, nella conoscenza del formato di pacchetti di rete che potrebbero evidenziare un tentativo d'intrusione.

Tuttavia, tralasciando quest'ultimo aspetto, appare evidente che possiamo accorgerci di un eventuale intrusione tenendo sotto controllo diversi componenti del sistema operativo, come quelli appena evidenziati. Il programma mostrato, ovviamente, non può essere considerato un IDS perfetto, ma dovrebbe offrire lo spunto per approfondire le tecniche di programmazione con WMI, sottolineando, ancora una volta, la potenza di questo linguaggio di scripting.

Francesco Lippo

Database Addio

Alla scoperta di PDO, un'estensione di PHP che rende semplice scrivere applicazioni totalmente indipendenti dal database utilizzato. Una tecnica comoda oltre che potente

Con la nuova versione di PHP, la 5.1, uno dei supporti che esisteva già in precedenza ovvero PDO è diventato un elemento fortemente integrato nella distribuzione. PDO è un meccanismo che consente di svincolarsi dal tipo di database usato e mette invece a disposizione un'interfaccia universale che consente di accedere, modificare e gestire i vari database con le stesse funzioni, indipendentemente dal db utilizzato.
È evidente che il vantaggio più immediato di questo genere di approccio è che qualunque tipo di piattaforma è supportata, e spostare un'applicazione da un db mysql a uno oracle comporta uno spreco di energie veramente basso, tipicamente limitato alla variazione della stringa di connessione.

I TUOI APPUNTI

Utilizza questo spazio per le tue annotazioni

INSTALLAZIONE

Se state utilizzando PHP 5.1 in ambiente Windows, l'installazione si riduce a decommentare la linea

```
extension=php_pdo.dll
```

all'interno del file php.ini. Sarà poi opportuno abilitare il supporto ai vari db che si intendono utilizzare

```
extension=php_pdo.dll
extension=php_pdo_firebird.dll
extension=php_pdo_mssql.dll
extension=php_pdo_mysql.dll
extension=php_pdo_oci.dll
extension=php_pdo_oci8.dll
extension=php_pdo_odbc.dll
extension=php_pdo_pgsql.dll
extension=php_pdo_sqlite.dll
```

sui sistemi unix bisogna compilare php con il supporto a pdo, nel modo seguente

```
./configure --with-zlib –enable-pdo=shared
```

e abilitare l'estensione, decommentando come al solito la corrispondente linea nel php.ini

```
extension=pdo.so
```

Se state usando un PHP inferiore al 5.1 sotto windows dovete consultare la PECL all'indirizzo pecl.php.net e scaricare il pacchetto corrispondente. In ambiente unix è sufficiente un

```
pecl install pdo
```

I DRIVER SUPPORTATI

Attualmente PDO supporta i seguenti formati di database

PDO_DBLIB	FreeTDS / Microsoft SQL Server / Sybase
PDO_FIREBIRD	Firebird/Interbase 6
PDO_MYSQL	MySQL 3.x/4.x
PDO_OCI	Oracle Call Interface
PDO_ODBC	ODBC v3 (IBM DB2, unixODBC and win32 ODBC)
PDO_PGSQL	PostgreSQL
PDO_SQLITE	SQLite 3 and SQLite 2

CONNETTERSI A UN DB TRAMITE PDO

È necessario creare un nuovo oggetto PDO a cui passare nel costruttore il driver, l'host, il database e le credenziali di autenticazione. Un esempio è il seguente:

```
<?php
$user="root";
$pass="unapassword";

$dbh = new PDO('mysql:host=
        localhost;dbname=iopdemo', $user, $pass);
?>
```

per verificare la corretta riuscita della con-

Conoscenze richieste
Basi di PHP

Software
PHP, IIS o Apache, MySQL o altri DB

Impegno

Tempo di realizzazione

nessione viene messo a disposizione un oggetto di tipo PDOException. Possiamo utilizzarlo nel modo seguente:

```
try
{
    $dbh = new PDO('mysql:host=
         localhost;dbname=iopdemo', $user, $pass);
}
catch (PDOexception $e)
{
    print "Errore!: " . $e->getMessage() . "<br/>";
   die();
}
```

Se l'errore non è gestito, tipicamente PHP mostrerà a schermo mostrando il "trace" dell'errore in modo standard. Questo ovviamente è sconveniente in quanto potrebbero venire mostrate informazioni che invece devono rimanere riservate.

CONNESSIONI PERSISTENTI

Un ulteriore caratteristica interessante, di PDO è quella di poter utilizzare delle connessioni persistenti. Tipicamente dopo che uno script è stato eseguito, la connessione al database viene chiusa, e riaperta quando lo script viene richiamato una seconda volta. Passando alcuni parametri a PDO è possibile mantenere la connessione persistente, ovvero la connessione non viene chiusa, i dati possono essere mantenuti in una cache e riutilizzati quando lo script viene richiamato una seconda volta. Un esempio valido di quanto appena detto è il seguente:

```
try
{
   $dbh = new PDO('mysql:host=localhost;dbname=
                   iopdemo', $user, $pass, array(
   PDO::ATTR_PERSISTENT => true));
}
catch (PDOexception $e)
{
   print "Errore!: " . $e->getMessage() . "<br/>";
   die();
}
```

In realtà l'ultimo parametro all'interno del quale abbiamo inserito un valore tale per cui la connessione rimane persistente è un array. Possiamo utilizzare questo array per passare a PDO valori particolarmente significativi che possono personalizzare il comportamento della libreria. Ad esempio:

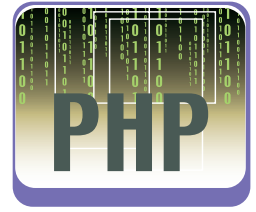
```
try
{
   $dbh = new PDO('mysql:host=localhost;dbname=
                   iopdemo', $user, $pass, array(
                   PDO::ATTR_PERSISTENT =>
                   true,PDO::ATTR_TIMEOUT=>
                   10,PDO::CASE_LOWER=>true));
}
catch (PDOexception $e)
{
   print "Errore!: " . $e->getMessage() . "<br/>";
   die();
}
```

in questo modo abbiamo informato PDO di utilizzare una connessione persistente, abbiamo settato il massimo timeout di connessione verso il server mysql in 10 secondi, abbiamo richiesto che tutti i nomi dei nomi delle colonne del db vengano convertiti in lettere minuscole.
Allo stesso modo alcuni di questi attributi sono accessibili in sola lettura, ed è possibile ottenerne il valore tramite una sintassi del tipo:

```
print($dbh->getAttribute(
                   PDO::ATTR_DRIVER_NAME));
```

attenzione, non tutti i driver supportano gli attributi messi a disposizione da PDO.

IL MODELLO TRANSAZIONALE

Molti di voi avranno sentito parlare di transazioni. Il concetto di transazione è abbastanza complesso e tralasceremo di discuterne in questo articolo, semplicemente ne diamo una nozione abbastanza semplificata affinché sia comprensibile come PDO gestisca questi aspetti. Diremo una transazione un'insieme di operazioni compiute su un database e tali che affinché la transazione sia valida tutte le operazioni che la compongono devono essere eseguite correttamente, in mancanza di questa condizione lo stato del db deve tornare alla condizione originale prima dell'inizio della transazione.
Le transazioni agiscono in modo che l'utente che sta effettuando un'operazione detiene un lock esclusivo sull'insieme di risorse che è coinvolto in una transazione. Questo avviene per evitare ad esempio due update simultanei di un dato, cosa che provocherebbe un'inconsistenza del database.

Per quanto riguarda PDO l'utilizzo delle transazioni risulta leggermente complicato dal fatto che non tutti i database supportano le transazioni, così PDO usa un meccanismo di *auto_commit*. Se l'operazione è singola PDO utilizzerà una transazione se il database la supporta, viceversa non lo farà. Se si vogliono utilizzare le transazioni in modo esplicito è necessario utilizzare le istruzioni *BeginTransaction* e *Commit*.
Con la prima inizieremo fisicamente la transazione, con il *commit* eseguiremo la transazione come se fosse una singola operazione, se qualcosa andasse storto potremmo utilizzare un meccanismo di *rollback*. Facciamo un esempio:

```
try
{
    $dbh = new PDO('mysql:host=localhost;dbname=
                    iopdemo', $user, $pass, array(
                    PDO::ATTR_PERSISTENT =>
                    true,PDO::ATTR_TIMEOUT=>
                    10,PDO::CASE_LOWER=>true));
}
catch (PDOexception $e)
{
    print "Errore!: " . $e->getMessage() . "<br/>";
    die();
};
try
 {
    $dbh->beginTransaction();
    $dbh->exec("insert into prova (id, nome,
              cognome) values (23, 'Pippo', 'Pluto')");
    $dbh->exec("insert into prova (id, nome,
          cognome) values (24, 'Paolino','Paperino')");
    $dbh->commit();

}
catch (Exception $e)
{
   $dbh->rollBack();
   echo "Failed: " . $e->getMessage();
}
```

In teoria questa transazione dovrebbe restituire un errore, primo perché la nostra tabella mysql ISAM non gestisce le transazioni, secondo perché non esiste una tabella prova all'interno del nostro db.
In realtà se provate questo spezzone di codice vi accorgerete che non viene sollevata nessuna eccezione, per gestire il problema dovremo utilizzare:

```
$dbh->setAttribute(PDO::ATTR_ERRMODE,
                   PDO::ERRMODE_EXCEPTION);
```

QUERY PREPARATE

Un concetto molto interessante di PDO è relativo all'uso dei così detti *Preparad Statements*. Si tratta di un concetto molto caro a parecchi linguaggi di alto livello. In sostanza viene consentito di inizializzare un modello di query.
All'interno di questo modello accanto ai tradizionali statement SQL vengono inseriti dei parametri, che poi verranno sostituiti al momento opportuno dai valori corretti.
Al solito il modo migliore per capire quello di cui stiamo parlando è fornire un esempio:

```
$stmt = $dbh->prepare("INSERT INTO istat
          (Comune, Dato) VALUES (:Comune, :Dato)");
$stmt->bindParam(':Comune', $comune);
$stmt->bindParam(':Dato', $dato);

$comune = 'Roma';
$dato = 106;
$stmt->execute();

$comune = 'Milano';
$dato = 02;
$stmt->execute();
```

Come vedete il concetto è molto semplice. Abbiamo inizialmente preparato la query, utilizzando due parametri che sono preceduti dal simbolo dei due punti. I due parametri in questione vengono poi collegati *(bindati)* alle variabili *$comune* e *$dato*. Ogni volta che si esegue un *"execute"*, i parametri vengono sostituiti con quelli a cui sono bindati.
Allo stesso modo è possibile utilizzare dei parametri posizionali per preparare lo statement, come segue:

```
$stmt = $dbh->prepare("INSERT INTO istat
                   (Comune, Dato) VALUES (?, ?)");
```

e poi bindarli con una sintassi del tipo

```
$stmt->bindParam(1, $comune);
$stmt->bindParam(2, $dato);
```

dove uno e due rappresentano le posizioni che nella query sono relative ai punti interrogativi.

FETCH DEI DATI CON PARAMETRI

Ovviamente gli *statement prepared* possono essere utilizzati all'interno di una clausola *where*, l'esempio è abbastanza elementare:

```
$stmt = $dbh->prepare("SELECT * FROM istat
                    where Comune Like ? AND Dato > ?");

$stmt->bindParam(2, $value);
$value=1000;

if ($stmt->execute(array("A%"))) {

  while ($row = $stmt->fetch()) {
   print ('<pre>');
   print_r($row);
   print ('</pre>');
  }
}
```

Da notare che in questo caso abbiamo utilizzato una modalità mista. Uno dei due parametri è stato passato bindandolo in modo posizionale, l'altro è stato passato all'interno di un array, come parametro del metodo execute.
La scelta fra i due metodi dipende esclusivamente dal vostro stile di codice, anche se abbiamo verificato che nel passaggio come parametro, in determinati casi la conversione di tipo può generare qualche problema. Nel nostro caso il parametro veniva passato come stringa e non come intero, motivo per cui il confronto non restituiva esattamente i dati che noi ricercavamo.

GESTIONE DI IMMAGINI O CAMPI BLOB

Una caratteristica interessante di PDO è data dalla possibilità di poter utilizzare degli algoritmi predefiniti per inserire o recuperare campi di grandi dimensioni. Normalmente per campi di grandi dimensioni si intendono campi blob che possono contenere ad esempio delle immagini, ma anche altri tipi di informazione. Vediamo subito un esempio, la nostra form per l'upload sarà contenuta in un file upload.php e sarà così composta

```
<form action="index.php" method="post"
      enctype="multipart/form-data" name="form1">
  <label>prova
  <input type="file" name="filep">
  </label>
  <p>
    <label>Submit
    <input type="submit" name="Submit"
                                value="Submit">
    </label>
  </p>
</form>
```

mentre per il codice di gestione sarà il seguente:

```
if(is_uploaded_file($_FILES['filep']['tmp_name'])) {
            $fp = fopen($_FILES['filep']['tmp_name'],
                                                'rb');

            $stmt = $dbh->prepare("insert into images
                          (immagine, tipo) values (?,?)");

            $stmt->bindParam(1, $fp,
                                   PDO::PARAM_LOB);

            $stmt->bindParam(2,
                             $_FILES['filep']['type']);

            $stmt->execute();
}
```

si noti che il binding del parametro uno viene effettuato puntando direttamente al nome del file che contiene l'immagine e che allo stesso modo, viene passato un parametro PDO::PARAM_LOB che comunica a PDO appunto che quello che un campo di grandi dimensioni e deve essere gestito piu o meno come uno stream.
Allo stesso modo recuperare l'immagine dal database segue più o meno lo stesso meccanismo:

```
$stmt = $dbh->prepare("select immagine, tipo
                                   from images");
$stmt->execute();

$stmt->bindColumn(1, $dati, PDO::PARAM_LOB);

$stmt->bindColumn(2, $type, PDO::PARAM_STR, 256);

$stmt->fetch(PDO::FETCH_BOUND);

header("Content-Type: $type");

fpassthru($dati);
```

CONCLUSIONI

PDO è un'estensione piuttosto utile che da un lato elimina il problema del porting delle applicazioni da una piattaforma di database ad un'altra, dall'altro introduce alcuni costrutti interessanti che rendono semplificata la vita del programmatore PHP. Si tratta di un'estensione che nel tempo dovrà sicuramente essere utilizzata a fondo, se si vogliono scrivere applicazioni estremamente portabili anche dal punto di vista dei database.

Facciamo i PDF

Capita spesso di dover realizzare stampe di una certa qualità oppure report impaginati in modo preciso. Per questo è opportuno utilizzare il formato PDF ma come interfacciarlo per PHP

Esistono diversi modi per poter stampare un documento in formato PDF da PHP. Per questa volta abbiamo scelto di utilizzare una classe esterna chiamata EZPdf e disponibile all'indirizzo http://www.ros.co.nz/pdf/. I motivi di questa scelta sono semplici. Utilizzare una classe esterna significa che chiunque possieda uno spazio PHP su un server in hosting può includere nei propri documenti PHP la classe e tutto funzionerà. Non c'è bisogno di chiedere al vostro provider di installare o compilare nuove estensioni, tutto quello di cui avete bisogno è semplicemente la classe in questione. Per cui andiamo ad iniziare.

INSTALLAZIONE E PRIMI PASSI

Come già detto il vantaggio di utilizzare questo genere di soluzione è quello di poter usare facilmente la classe senza dover ricorrere a installazioni complicate. È sufficiente scompattare il file tar.gz all'interno di una sottodirectory originale del progetto e poi includerla con un normale include. Ad esempio:

```
<?
include_once("lib/class.ezpdf.php");
?>
```

A questo punto non ci resta che iniziare con un semplicissimo "Hello World". Il costruttore della classe è Cezpdf che risponde ad una sintassi del tipo

```
Cezpdf([paper='a4'],[orientation='portrait'])
```

Per cui il primo passo è istanziare un oggetto di tipo Cezpdf, come segue:

```
$pdf =& new Cezpdf();
```

a questo punto abbiamo a disposizione una sorta di lavagna bianca su cui scrivere, ma attenzione, nulla accadrà richiamando il file che contiene queste istruzioni. Di fatto la nostra lavagna è instanziata in memoria, ma non abbiamo ancora deciso come sarà il nostro output, potrebbe essere un PDF da visualizzare inline all'interno del browser, oppure potrebbe essere uno stream da salvare sull'HD. Proviamo ad esempio le istruzioni:

```
include_once("lib/class.ezpdf.php");
$pdf =& new Cezpdf();
$pdf->selectFont('lib/fonts/Helvetica.afm');
$pdf->ezText('Hello World!',50);
$pdf->ezStream();
```

oppure

```
include_once("lib/class.ezpdf.php");
$pdf =& new Cezpdf();
$pdf->selectFont('lib/fonts/Helvetica.afm');
$pdf->ezText('Hello World!',50);
$pdfcode = $pdf->output();
$fp=fopen('/tmp/prova.pdf','wb');
fwrite($fp,$pdfcode);
fclose($fp);
```

Avete già capito come funziona il meccanismo fondamentale di EZPdf. Si instanzia una "lavagna" in memoria, si scrive sulla lavagna, si decide come gestire l'output. Tutto quello che è necessario conoscere sta nel mezzo, ovvero i metodi per disegnare, scrivere, inserire un'interruzione di pagina, scrivere in grassetto e tutte le altre opzioni tipiche della scrittura di precisione.
Di due delle funzioni più importanti abbiamo già fatto la conoscenza, ovvero ezText che ci consente di scrivere un testo e SelectFont che ci consente di capire come scegliere i caratteri.

SCRIVERE IL TESTO

Come già detto il metodo basilare per scrivere il testo è ezText, risponde alla sintassi:

```
ezText(text,[size],[array options])
```

dove text, è il testo da scrivere, size è la dimensione del font, seguono poi una serie di opzioni formattate in un array, ad esempio:

```
<?
include_once("lib/class.ezpdf.php");
$pdf =& new Cezpdf();
$pdf->selectFont('lib/fonts/Helvetica.afm');
$primariga="Questo è un testo di prova per
                                        ioProgrammo";
$secondariga="disponiamolo su due righe per capire
come funziona la giustificazione o l'allineamento";

$options=array(
               'justification'=>'right',
               'left'=>'20'
               );
$pdf->ezText($primariga,12,$options);
$pdf->ezText($secondariga,12,$options);
$pdf->ezStream();
?>
```

Questo spezzone di codice è significativo di come prima venga selezionato un font e poi venga scritto il testo su due righe, utilizzando l'array di opzioni per settare la giustificazione e la distanza dal margine sinistro. E se volessimo scrivere in verticale?
In tal caso sarebbe necessario utilizzare il metodo Addtext che supporta una serie di parametri leggermente più complesso rispetto a exText. Il metodo AddText risponde a una sintassi del tipo:

```
addText(x,y,size,text,[angle=0],[adjust=0])
```

dove x,y rappresentano le coordinate da cui iniziare a scrivere, text rappresenta il testo da scrivere, angle l'angolo di inclinazione, e adjust viene utilizzato per settare il valore della distanza degli spazi. Ad esempio:

```
<?
include_once("lib/class.ezpdf.php");
$pdf =& new Cezpdf();
$pdf->selectFont('lib/fonts/Helvetica.afm');
$primariga="Questo è un testo di prova
                               per ioProgrammo";
$pdf->addtext(150,150,12,$primariga,45,1);
$pdf->ezstream();
?>
```

stampa un testo a 45 gradi nella pagina. Da notare che l'origine delle coordinate è settato nell'angolo in basso a sinistra.
Addtext supporta anche l'uso di codice inline del tipo:

```
$primariga="Questo è un testo di prova
               per<b>ioProgrammo</b>";
$pdf->addtext(150,150,12,$primariga,45,1);
$pdf->ezstream();
```

stampa la parola "ioProgrammo" in grassetto.

INSERIMENTO DELLE IMMAGINI

Utilizzare un'immagine all'interno del testo non segue una logica molto diversa da quella precedente, in questo caso il comando da utilizzare è addJpegFromFile oppure in alternativa addPngFromFile, la differenza fra i due comandi risiede ovviamente semplicemente nel formato del file da aggiungere.
Ambedue rispondono alla sintassi:

```
addJpegFromFile(imgFileName,x,y,w,[h])
```

oppure

```
addPngFromFile(imgFileName,x,y,w,[h])
```

dove imgFileName è il nome del file da inserie, x,y rappresentano le coordinate, w ed h rappresentano rispettivamente l'altezza e la larghezza dell'immagine. Un esempio di codice è il seguente:

```
<?
include_once("lib/class.ezpdf.php");
$pdf =& new Cezpdf();
$img="lib/ros.jpg";
$pdf->addJpegFromFile($img,100,100,50);
$pdf->ezstream();
?>
```

Volendo, invece di inserire immagini precostruite è possibile anche utilizzare forme disegnate ex novo. Prendiamo ad esempio il caso di un'ellisse
Il comando base è

```
ellipse(x0,y0,r1,[r2=0],[angle=0],[nSeg=8])
```

Dove x0 e y0 rappresentano le coordinate del centro. r1 e r2 i due raggi, angle l'angolo di inclinazione dell'ellisse, e nSeg il numero di curve di Bezier in cui il disegno viene riprodotto. Il numero di curve in questione rappresenta ovviamente anche l'accuratezza del grafico, normalmente 8 consente una buona approssimazione. L'esempio di codice è il seguente:

```
<?
include_once("lib/class.ezpdf.php");
$pdf =& new Cezpdf();
$img="lib/ros.jpg";
$pdf->ellipse(200,300,30,0,0,8);

$pdf->ezstream();
?>
```

notate che ponendo il secondo raggio uguale a zero abbiamo ottenuto il disegno di un cerchio e non di un'ellisse. Per ottenere quanto desiderato è sufficiente variare il secondo parametro.

Utilizza questo spazio per le tue annotazioni

IO PROGRAMMO BY EXAMPLE

IMPARA A PROGRAMMARE IN MODO PRATICO E DIVERTENTE, CON GLI ESEMPI PASSO PASSO CHE TI GUIDANO ALLA COSTRUZIONE DEL CODICE

VUOI INVIARE UN ESEMPIO?

Se sei un programmatore esperto ed hai risolto un problema, puoi aiutare gli altri pubblicando il tuo codice. Proponi i tuoi esempi scrivendo a ioprogrammo@edmaster.it

COSTRUIRE UN FILTRO PER I DATI

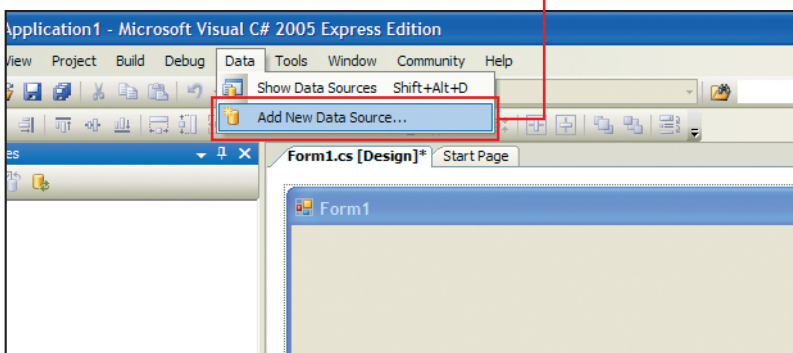
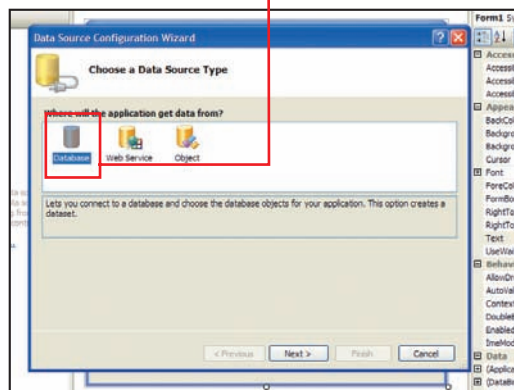
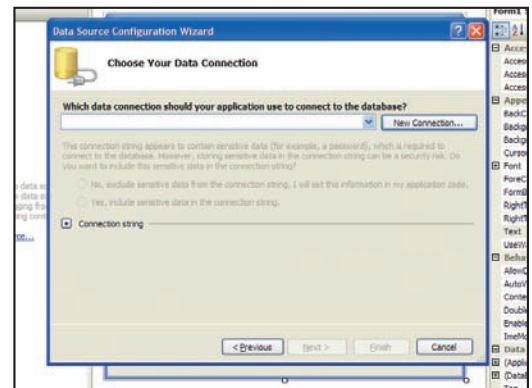
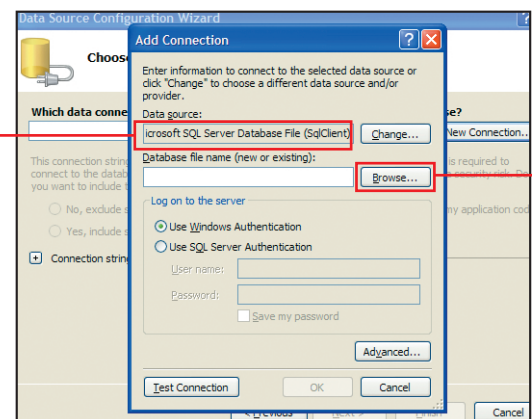
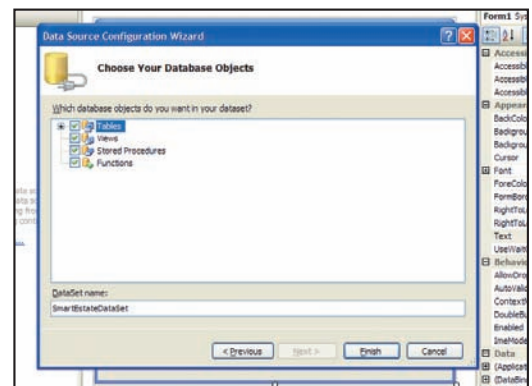
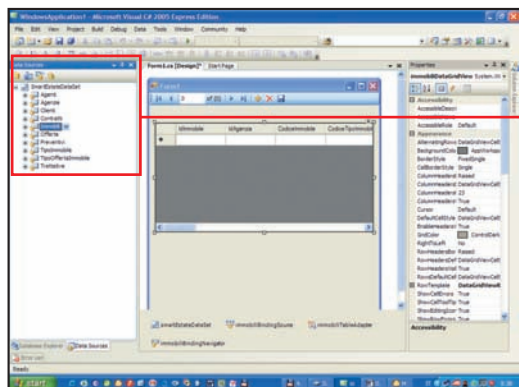
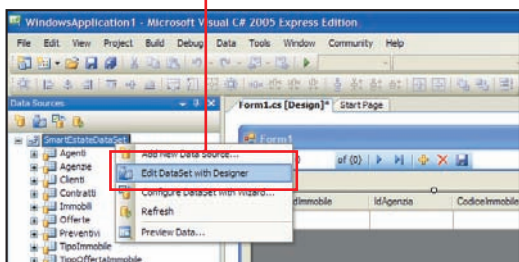
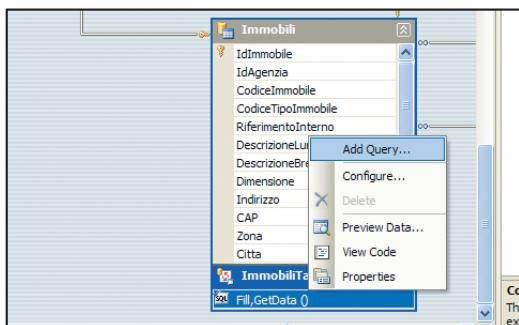
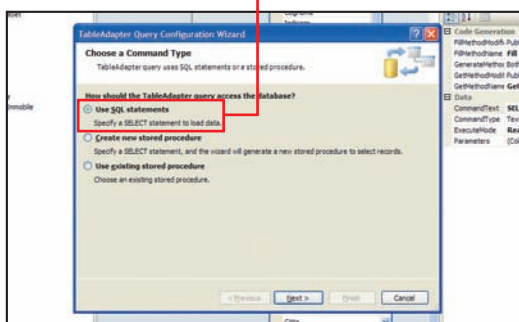
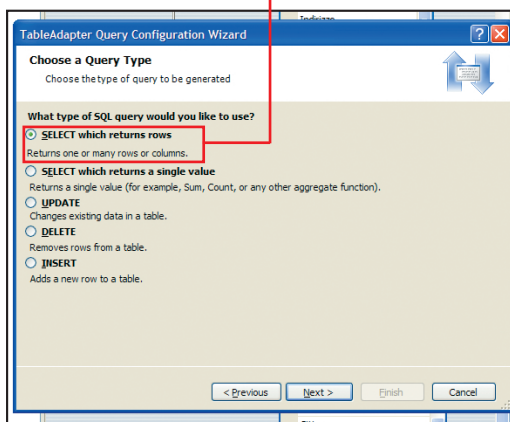
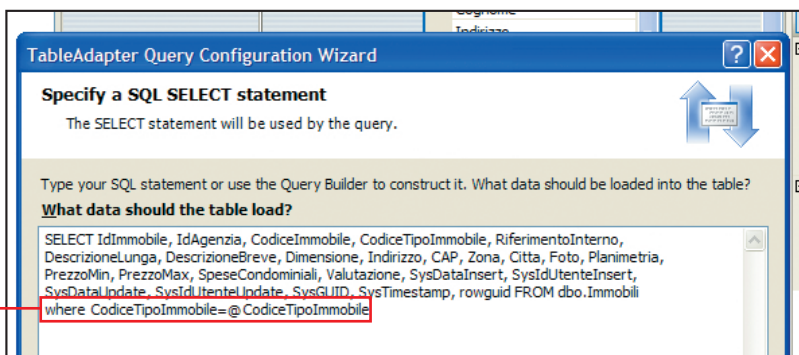
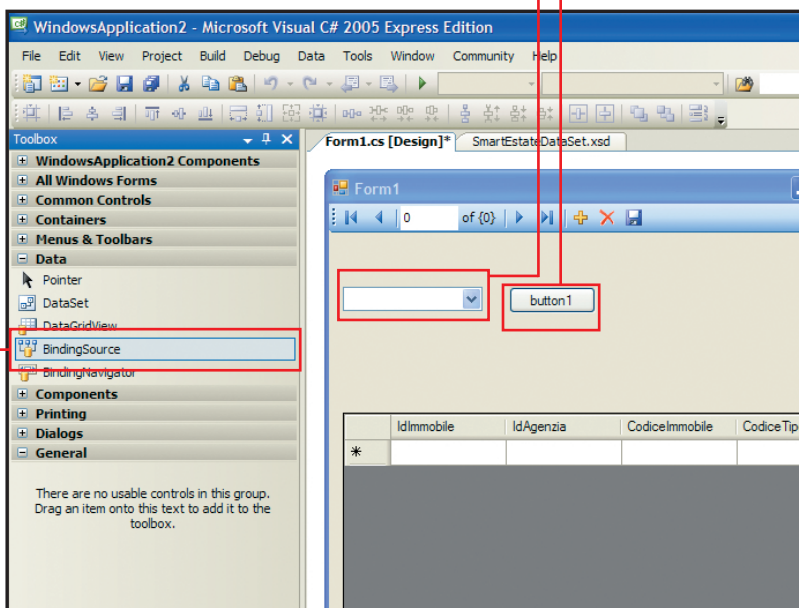
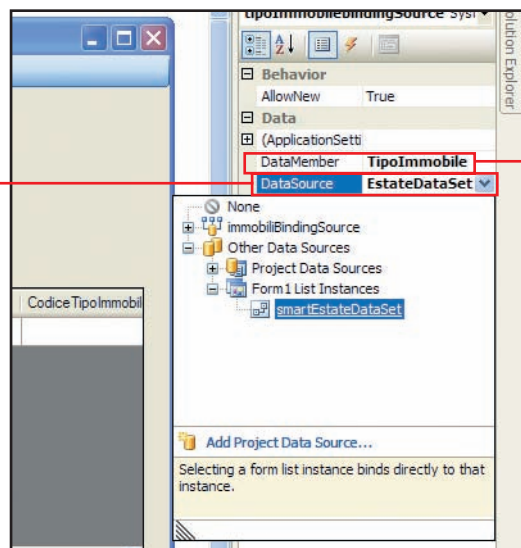
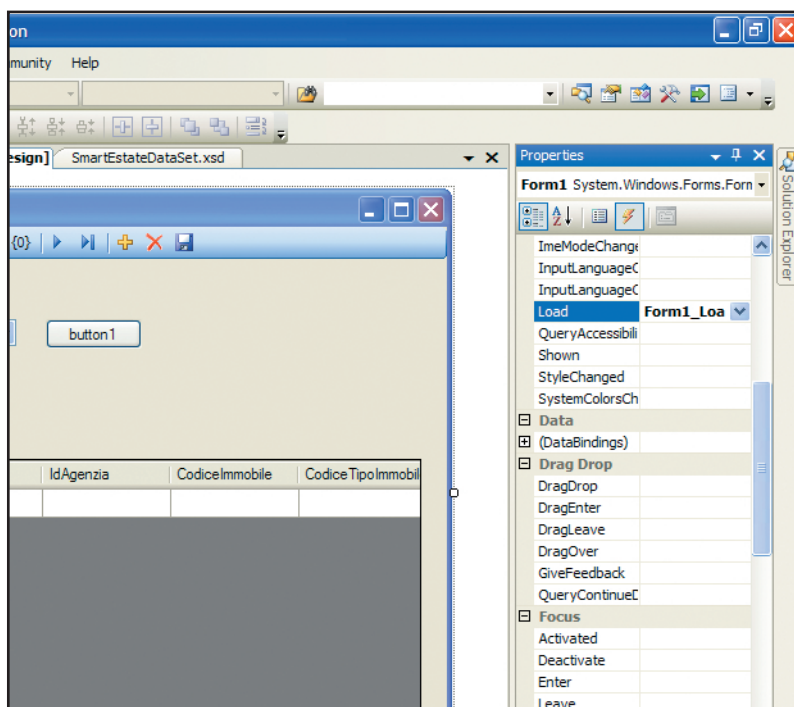
DISPONIAMO DI UNA TABELLA E DI UN MENU A TENDINA. VOGLIAMO CHE IL CONTENUTO DELLA TABELLA VARI A SECONDA DEL DATO SELEZIONATO NEL MENU, COME FARE?

C#

VISUAL BASIC.NET

Per il nostro esempio utilizzeremo un database Sql Server 2005 ipotizzando che esista una tabella di immobili contenente un elenco di appartamenti in vendita, e una tabella tipoimmobili che contiene un elenco di stringhe che identificano un tipo di immobile. Ovviamente il nostro filtro vuole che in una griglia siano visualizzati solo i tipi di immobile che riteniamo più opportuni. Per chi volesse provare il tutto, abbiamo utilizzato un database chiamato SmartEstate, reso disponibile da Fabio Santini, uno dei guru di Microsoft sul suo sito www.fsantini.net. All'interno dello stesso sito trovate l'applicazione SmartEstate utilizzata proprio per presentare Visual Studio 2005 nel corso di molti degli eventi di formazione ufficiale. L'esempio che vi proponiamo qui è presente anche all'interno della stessa applicazione riprodotto interamente in Visual Basic.net.

FACCIAMOLO IN C#

1 Dal menu *data* selezioniamo *"Add New Data Source"*

2 Nel wizard a seguire scegliamo *"Database"* e poi clicchiamo su *Next*

3 Scegliamo *New Connection* per stabilire la connessione al database

4 Lasciamo come *"DataSource" Microsoft Sql Server* e clicchiamo su *Browse* per individuare il file .MDF che contiene il Database di SQL Server. Ovviamente se il nostro Db fosse access sceglieremmo come data source *"Access"* e cercheremmo il corrispondente file .MDB. Nelle schermate successive clicchiamo su *Next* e poi su *"yes"* fino alla fine.

5 Scegliamo di importare tutto il contenuto del database all'interno del datasource, selezionando i segni di spunta per tute le voci. Poi clicchiamo su *"Finish"*

6 Dalla finestra *"DataSources"* trasciniamo sulla form l'intera tabella di cui vogliamo ottenere una rappresentazione a griglia

7 Con il tasto destro del mouse clicchiamo sul dataset e selezioniamo *"Edit DataSet With Designer"*

8 Scegliamo la tabella che vogliamo filtrare e cliccando con il tasto destro del mouse scegliamo *"Add Query"*

9 Selezioniamo *"Use Sql Statements"* e andiamo avanti

10 Scegliamo *"Select which return rows"* e andiamo avanti

11 Modifichiamo la query aggiungendo la condizione *'where CodiceTipoImmobile=@CodiceTipoImmobile'*

12 Trasciniamo sulla form un *bottone* e un *combobox*. Inoltre aggiungiamo un componente di tipo *BindingSource*, a cui cambieremo nome e chiameremo *"tipoImmobilebindingSource"*

13 Nella finestra delle proprietà assicuriamoci che il *datasource* punti al *dataset* che stiamo usando nel nostro programma e che il *datamember* punti alla tabella *"tipoimmobile"*

14 Clicchiamo due volte nell'evento *On_Load* della form per generare il template di gestione del codice

15 Il codice da inserire nel template è il seguente:

```
foreach (SmartEstateDataSet.TipoImmobileRow row
         in smartEstateDataSet.TipoImmobile )
{
  comboBox1.Items.Add(row.CodiceTipoImmobile);
}
```

16 Infine clicchiamo due volte sul bottone per generare il template dell'evento *OnClick* e aggiungiamo il codice:

```
private void button1_Click(object sender, EventArgs e)
{
   this.immobiliTableAdapter.FillBy(
      smartEstateDataSet.Immobili,comboBox1.Text);
}
```

COME FUNZIONA

Tutta la prima parte fino al punto 5 consente di stabilire una connessione al database. Inserendo la griglia sulla form vengono automaticamente crati un BindingSource e un TableAdapter per mostrare i dati in questione. Il BindingSource "immobilibindingsource" aggancia la tabella immobili sul database. Il TableAdapter "immobiliTableAdapter" è una sorta di contenitore che contiene i dati recuperati tramite una query dalla tabella "Immobili". A fare da tramite è il Binding Source di cui abbiamo già parlato.
Tramite il Designer abbiamo creato un nuovo metodo fillby che esegue una query parametrica sulla tabella "immobili" e riempie il corrispondente immobiliTableAdapter che modifica il contenuto della griglia.
Abbiamo poi aggiunto un nuovo componente BindingSource questa volta agganciato alla tabella "TipoImmobile". Nel momento in cui la form viene caricata, tramite l'evento OnLoad riempiamo il combobox con i valori presi dalla tabella "TipoImmobile". Per farlo sfruttiamo il BindingSource che abbiamo precedentemente aggiunto. Infine cliccando sul bottone non facciamo altro che eseguire il metodo FillBy che avevamo creato con il designer e di conseguenza modifichiamo il tableadapter della tabella immobili riempiendo con i valori filtrati la griglia.

FACCIAMOLO IN VISUAL BASIC .NET

I punti dall'uno al 14 sono identici.

1 Il codice da inserire per la gestione dell'evento onload della form diventa:

```
Me.ImmobiliTableAdapter.FiltraTipoImmobile(
     Me.SmartEstateDataSet.Immobili, combobox1.Text)
```

2 Per l'evento *OnClick* sul bottone invece abbiamo

```
Me.ImmobiliTableAdapter.fillby(Me.SmartEstateData
                    Set.Immobili, combobox1.Text)
```

UN FILTRO PER I DATI IN ASP.NET 2.0

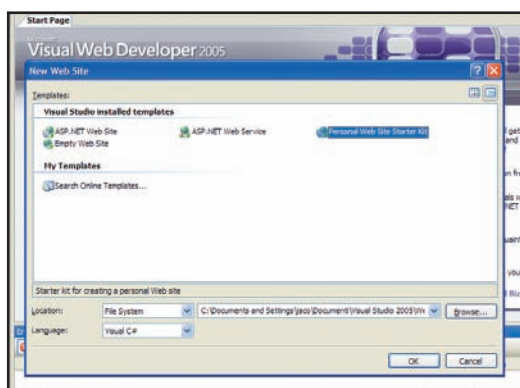
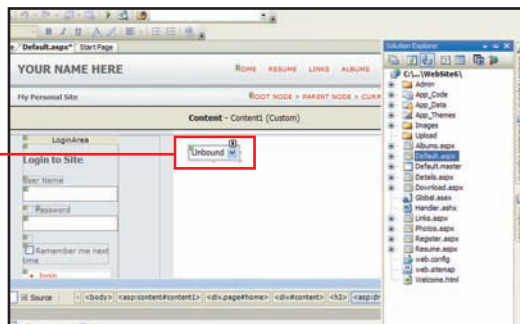
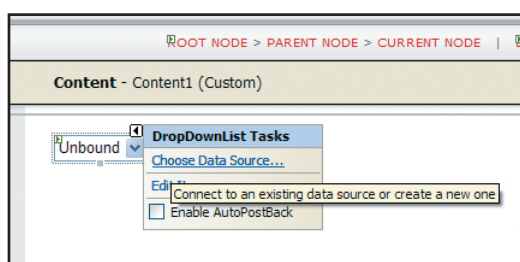
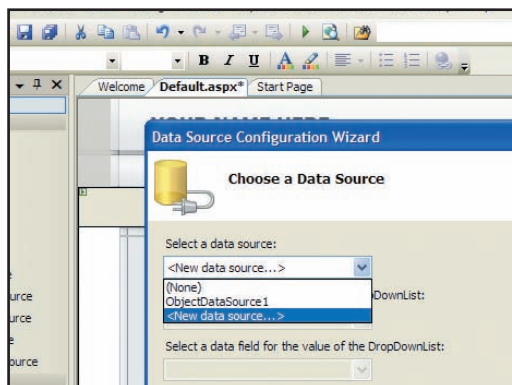
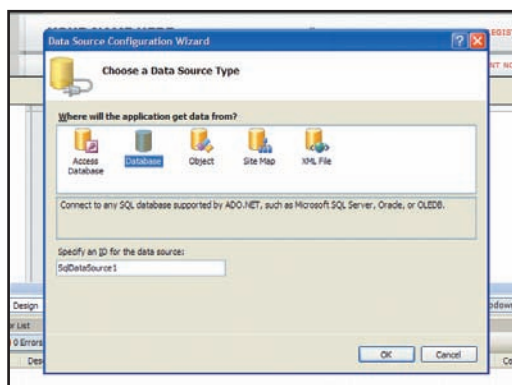
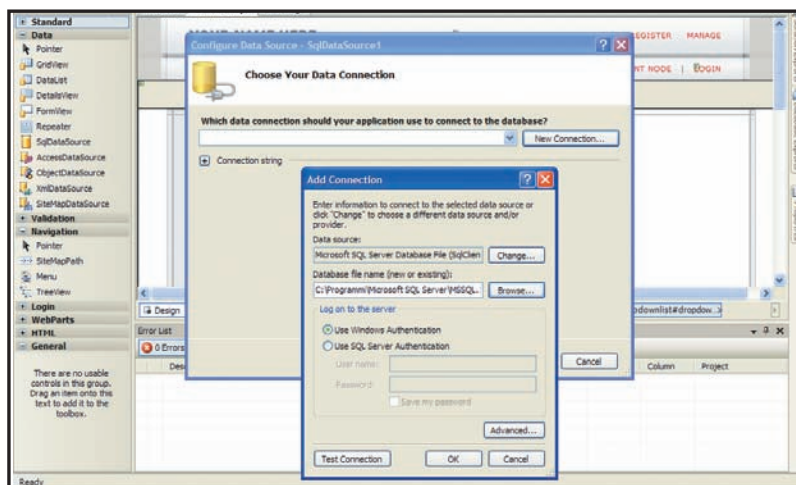
COME POSSO SELEZIONARE I DATI DI UNA TABELLA TRAMITE UN COMBOBOX?

C#

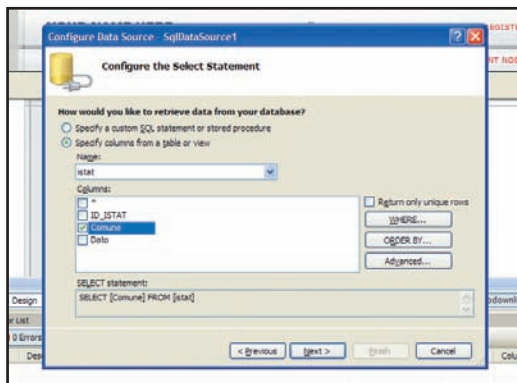
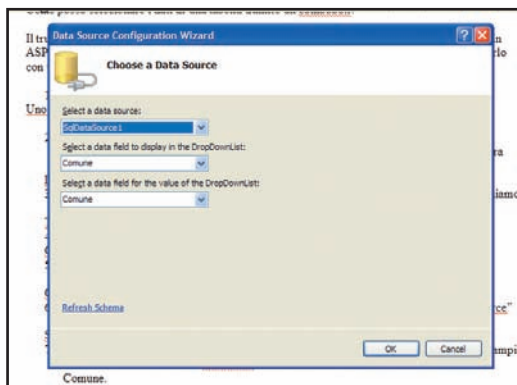
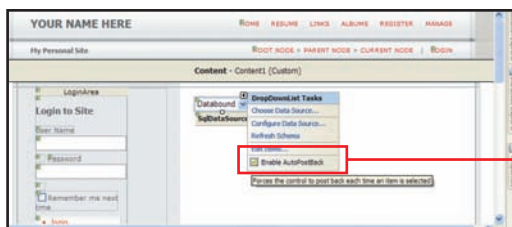
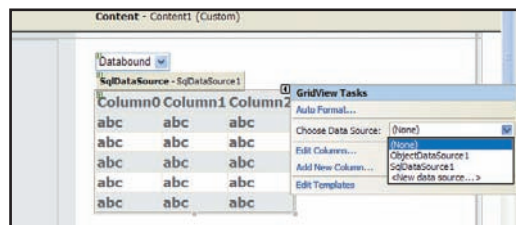
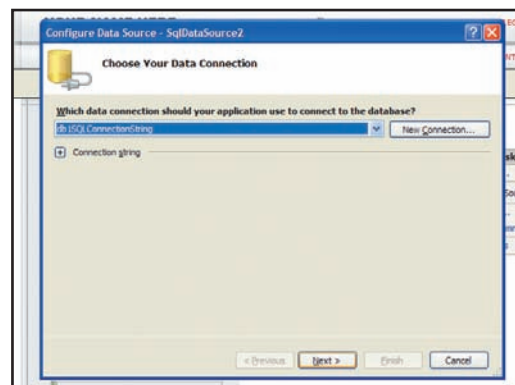
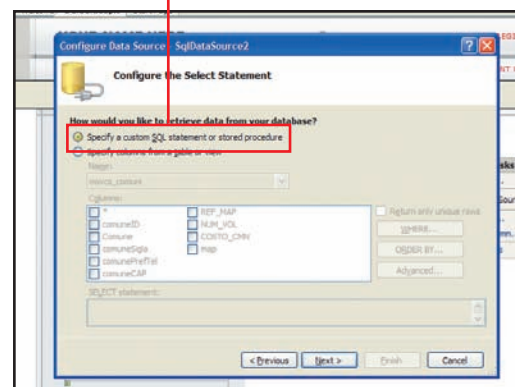
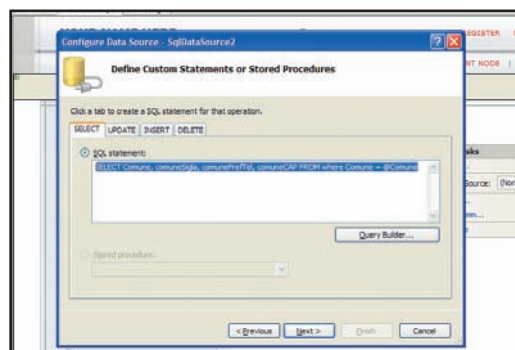
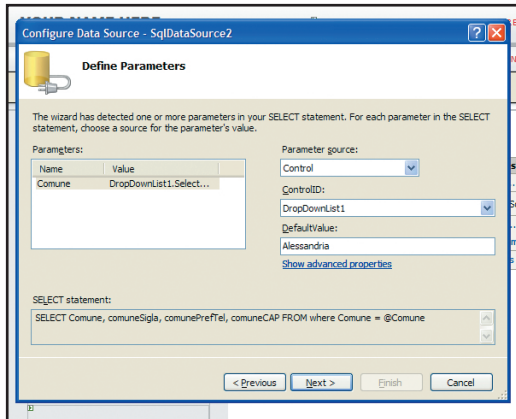
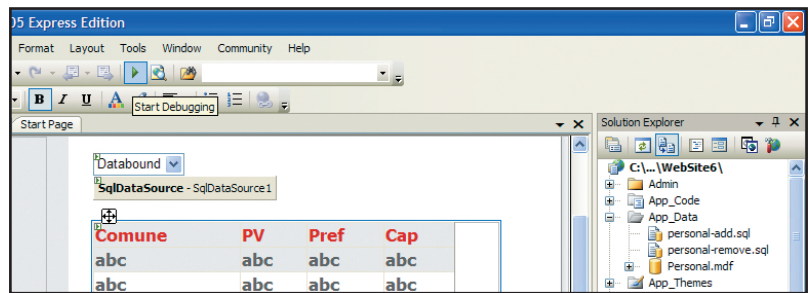
VISUAL BASIC.NET

Il trucco è praticamente identico a quello precedente mostrato per Visual Basic.NET e C#, ma in ASP.NET 2.0 il procedimento è leggermente diverso, motivo per cui abbiamo preferito illustrarlo con un miniarticolo a sé stante.

FACCIAMOLO IN C#

1 Iniziamo creando un *"New Web Site"* e selezionando *"Personal Web Site Starter Kit"*.

2 Nel *solution Explorer* selezioniamo *Default .aspx* e in basso a sinistra portiamoci nella modalità *"Design"*. Sostituiamo il contenuto della pagina d'esempio aggiungendo per ora solo un *combobox*

3 Clicchiamo sullo *SmartTag* in alto a destra nel combobox e dal menu a tendina selezioniamo *"Choose Data Source"*

4 Nella dialog box che segue scegliamo *"New Data Source"*

5 Per questa demo prenderemo i dati da un db Sql Server 2005, quindi selezioniamo *"Database"*

6 Scegliamo *"New Connection"* e nell'interfaccia che segue indichiamo come *"Data Source"* SQL Server e usiamo il tasto *"Browse"* per puntare al file che contiene i dati di SQL

DEBUG DI APPLICAZIONI IIS

Visual Studio 2005 ha un Web Server integrato per cui non è più necessario, come succedeva in precedenza, dover installare IIS, bensì è sufficiente lanciare l'applicazione direttamente dall'ambiente di Visual Studio. Su una porta libera verrà lanciato il Web Server, l'applicazione sarà compilata e la vedrete in esecuzione.

7 Clicchiamo su Next fino a quando non giungiamo alla schermata che ci chiede quali campi devono essere contenuti nel combobox. Nel nostro caso selezioniamo il nome di un Comune.

8 Stabiliamo che nel combobox deve essere mostrato il nome di un Comune e che il valore del combo sarà il nome del Comune stesso

9 Infine abiltiamo il *"PostBack"* sul controllo combobox

10 Trasciniamo una DataGrid sulla pagina, e come sempre dallo SmartTag selezioniamo "New Data Source"

11 Selezioniamo nuovamente *"DataBase"* e nella dialog box che segue dal menu a tendina scegliamo la connection string che avevamo già settato con i passi precedenti: *"db1-SQLconnectionString"*.

12 Questa volta scegliamo *"Specify a custom SQL statement or stored procedure"*

13 Creiamo la query che ci interessa ricordandoci di aggiungere la clausola *where* con un parametro

14 Nella form che segue indichiamo che il parametro da passare alla query deve essere prelevato da un controllo, nella fattispecie dal combobox che avevamo creato in precedenza. Diamo anche un valore di default di modo che non ci siano inconsistenze nei dati

15 Per Debuggare l'applicazione clicchiamo sul bottone in alto nella toolbar

COME FUNZIONA

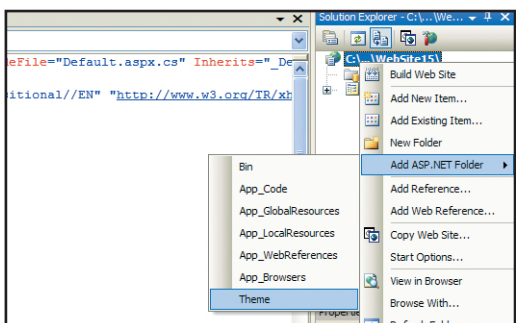
Con i passi da 1 a 10 riempiamo il combobox con un valore preso da una query. Creiamo poi una datagrid che prende i dati da una query parametrica. La nuova versione di ASP.NET consente di prendere i parametri da passare ad una query direttamente dal contenuto di un controllo posizionato su una pagina. È esattamente quello che abbiamo fatto noi. Grazie al postback ogni volta che il contenuto del combobox cambia, verrà di nuovo richiamata la query che riempie la datagrid e sarà inviato il parametro corretto.

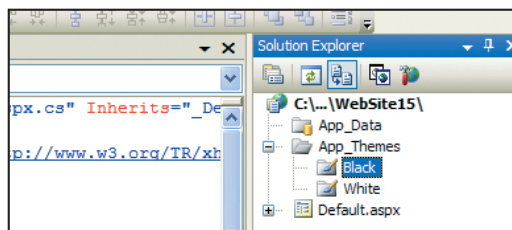
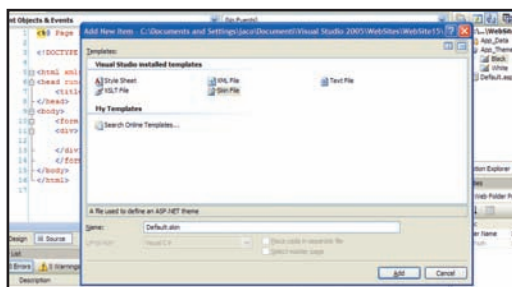
COME POSSO CREARE UN TEMA PER IL MIO SITO WEB?

CON ASP.NET MODIFICARE DINAMICAMENTE L'ASPETTO DI UN SITO È UN'OPERAZIONE PIUTTOSTO SEMPLICE, IMPARIAMO A USARE GLI SKIN

FACCIAMOLO CON ASP.NET

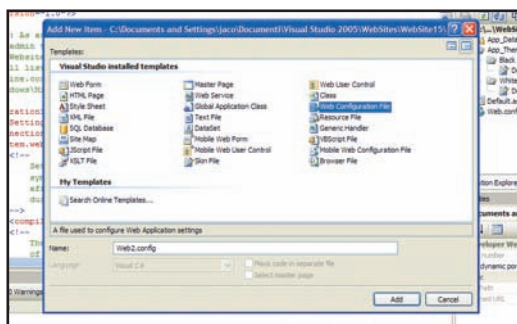
ASP.NET

1 Dopo avere creato un nuovo sito web clicchiamo con il tasto destro del mouse sull'icona che rappresenta il progetto nel solution explorer e dal menu contestuale "Add Asp.NET folder/Theme"

2 Chiamiamo il nuovo tema ad esempio White

3 Ripetiamo l'operazione e aggiungiamo anche il tema Black

4 Clicchiamo prima sul tema "White" con il tasto destro del mouse e dal menu a tendina scegliamo "Add/New Item", dalla dialog box che compare scegliamo "Skin File" e diamogli nome "Default

.Skin”. Ripetiamo l’operazione per il tema White

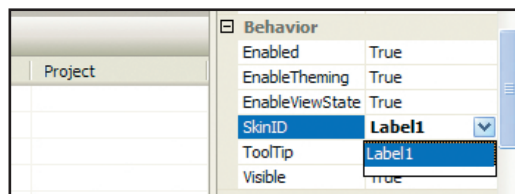
5 Nei due file appena creati aggiungiamo rispettivamente le seguenti linee

```
<asp:Label  runat="server" SkinId="Label1"
            BackColor="#000000" ForeColor="#ffffff">
                                   Label</asp:Label>
<asp:Label  runat="server" SkinId="Label1"
            BackColor="#990000" ForeColor="#ffffff">
                                   Label</asp:Label>
```

6 Di nuovo clicchiamo con il tasto destro del mouse e di nuovo scegliamo “Add/New Item” dalla dialog box che compare selezioniamo “Web Configuration File”

7 Nel file appena creato aggiungiamo, all’interno del nodo “System.Web”

```
<pages styleSheetTheme="White"/>
```

oppure Black a seconda del tema che usato.

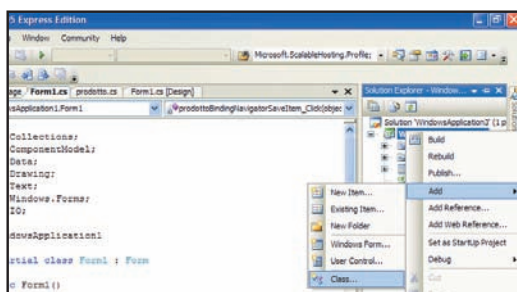
8 Trasciniamo una label sulla pagina e nella sua proprietà SkinID settiamo “Label1” riferendoci all’id che avevamo dato alla label nel file di skin

COME FUNZIONA

Nel file skin definiamo l’aspetto estetico della label. Possiamo definire un particolare aspetto per un particolare tipo di label identificando la tipologia tramite lo SkinID. Il file Web.Config fa da ponte tra il file di skin e l’applicazione. La property SkinID del componente può essere settata per essere associata alla particolare tipologia di label “identificata” nel file di skin.

COME POSSO CREARE UNA MIA CLASSE PER LA GESTIONE DEI DATI?

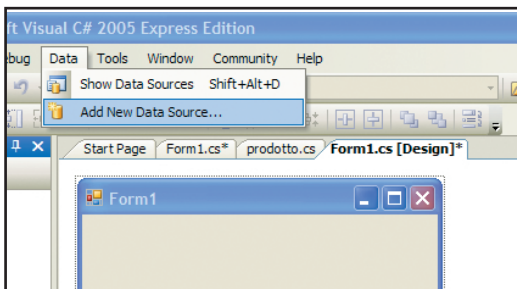
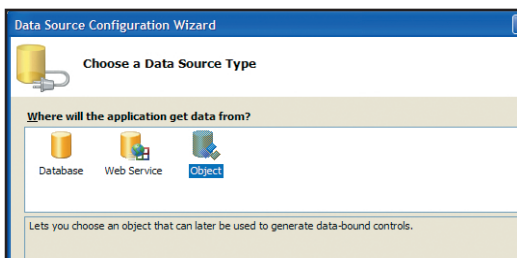
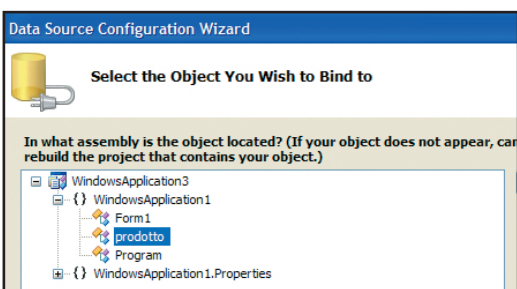
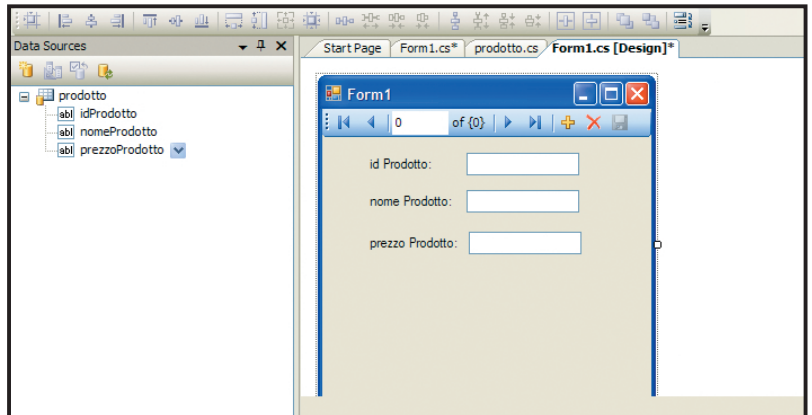
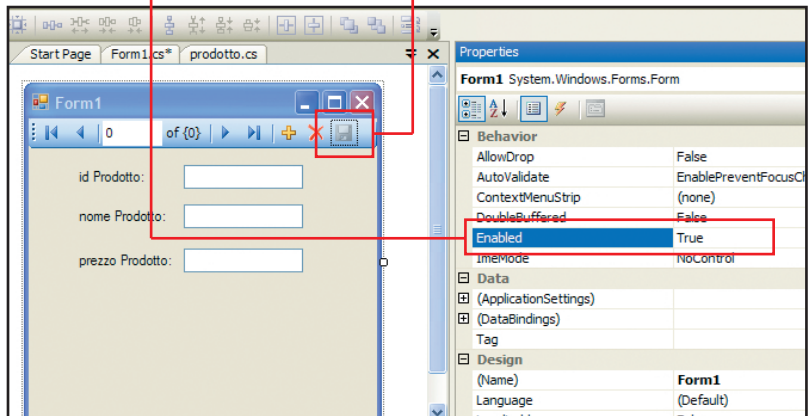
IN ALCUNI CASI I PROVIDER MESSI A DISPOSIZIONE DA VISUAL STUDIO 2005 POTREBBERO ESSERE INSUFFICIENTI, OPPURE ABBIAMO NECESSITÀ DI GESTIRE UN NOSTRO PARTICOLARE FORMATO. VEDIAMO COME INTERFACCIARE I MECCANISMI DI AUTOMAZIONE DI VISUAL STUDIO AD UNA NOSTRA CLASSE

FACCIAMOLO IN C#

1 Prima di tutto portiamoci nel solution explorer e aggiungiamo una nuova classe che si chiamerà *“prodotto”* e sarà contenuta nel file *prodotto.cs*.

2 La classe deve derivare dall’interfaccia *INotifyPropertyChanged*. Il nostro esempio propone la seguente implementazione:

```
class prodotto:INotifyPropertyChanged
 {
     private string IDProdotto;
     private string NOMEProdotto;
     private string PREZZOProdotto;
     #region INotifyPropertyChanged Members
     public event PropertyChangedEventHandler
                                    PropertyChanged;
     #endregion
     public string idProdotto {
         get {
             return IDProdotto;
         }
         set {
             IDProdotto = value;
         }
     }
     public string nomeProdotto {
         get
```

```
            {
                return NOMEProdotto;
            }
            set
            {
                NOMEProdotto = value;
            }
        }
        public string prezzoProdotto {
            get
            {
                return PREZZOProdotto;
            }
            set
            {
                PREZZOProdotto = value;
            }
        }
    }
```

3 Clicchiamo su *Data/Add new Data Sources* per aggiungere una sorgente dati

4 Dalla maschera che ci viene proposta selezioniamo *"Object"*

5 Scegliamo nella gerarchia che ci viene proposta la classe *"Prodotto"* che abbiamo creato in precedenza.

6 Dalla finestra del datasources trasciniamo sulla form i campi che ci interessano. Notate che rispettano fedelmente quelli che abbiamo costruito nella nostra classe.

7 Portiamoci sulla form e selezioniamo dal *Navigator* il tasto con l'icona di un floppy che è associato al *salvataggio dei dati su disco*. Portiamo la sua proprietà *Enabled* a *True*

L'INTERFACCIA INOTIFYPROPERTYCHANGED

Viene utilizzata per notificare a un client che il valore di una proprietà è cambiato. Tipicamente un client potrebbe essere un componente BiningSource. È possibile anche modificare il comportamento dell'evento *PropertyChangedEvent Handler* che si scatena appunto quando un valore viene cambiato.

8 Infine clicchiamo due volte sull'evento *OnClick* del bottone *'Save'* per generare il template del codice che gestirà l'evento. Il codice di gestione potrebbe essere il seguente:

```
private void
        prodottoBindingNavigatorSaveItem_Click(
                        object sender, EventArgs e)
    {
        IList list = prodottoBindingSource.List;
        FileStream file = File.OpenWrite(
                                "prodotti.txt");
```

```
        StreamWriter wr = new StreamWriter(file);
        foreach (prodotto prod in list)
        {
            wr.WriteLine("{0};{1};{2}",
                prod.idProdotto , prod.nomeProdotto,
                                  prod.prezzoProdotto);
        }
        wr.Close();

    }
```

I NAMESPACE DA IMPORTARE IN C#

Perché tutto funzioni, nella classe prodotto devono essere inclusi i seguenti namespace:

```
using System;
using System.Text;
using System.ComponentModel;
```

Nella Form1 ci devono invece essere i seguenti NameSpace:

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.IO;
```

COME FUNZIONA?

La classe prodotto implementa l'interfaccia INotifyPropertyChanged cosa che la rende "Bindable" dal Wizard di Visual Studio 2005 come classe che rappresenta metodi proprietari per la gestione dei dati. Una volta che il Wizard ha 'bindato' la classe come Data Source dell'applicazione è facile trasportare i campi sulla form come si fa normalmente per un database.
A questo punto i vostri dati saranno presenti solo in memoria, per poterli trasferire su disco utilizziamo il bottone 'Save' posto sul Navigator. La pressione di tale bottone apre uno strema in lettura, e scansiona il "BindingSource" con un ciclo scrivendo su un file i dati separati da una virgola.

I NAMESPACE DA IMPORTARE IN VB

Nella classe prodotto aggiungere:

```
Imports System
        .ComponentModel
```

Nella classe Form1 aggiungere

```
Imports System.Collections
Imports System.IO
```

FACCIAMOLO IN VISUAL BASIC

1 Creiamo l'apposita classe prodotto, il procedimento è uguale alla versione per C#, il codice di gestione della classe è il seguente:

```
Imports System.ComponentModel

Public Class prodotto
    Implements INotifyPropertyChanged
    Private IDProdotto
    Private NOMEProdotto
    Private PREZZOProdotto

    Public Event PropertyChanged As
                 PropertyChangedEventHandler _
        Implements
            INotifyPropertyChanged.PropertyChanged

    Public Property id() As String
        Get
            Return Me.IDProdotto
        End Get

        Set(ByVal value As String)
            Me.IDProdotto = value
        End Set
    End Property

    Public Property nome() As String
        Get
            Return Me.NOMEProdotto
        End Get
        Set(ByVal value As String)
            Me.NOMEProdotto = value
        End Set
    End Property

    Public Property prezzo() As String
        Get
            Return Me.PREZZOProdotto
        End Get
        Set(ByVal value As String)
            Me.IDProdotto = value
        End Set
    End Property

End Class
```

2 Ripetiamo i passi dal 2 all'otto esattamente come abbiamo fatto per C#. E scriviamo nuovamente il codice di gestione dell'evento *Click* in relazione al bottone *'Save'*, che diventa:

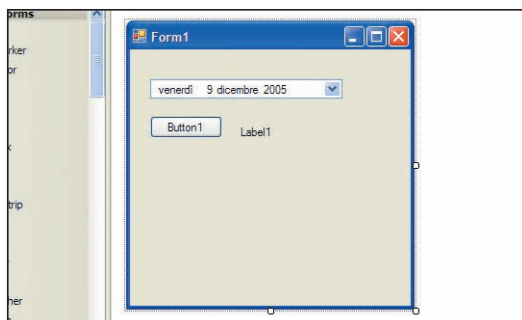
```
Public Class Form1

    Private Sub ProdottoBindingNavigatorSaveItem_
        Click(ByVal sender As System.Object, ByVal
              e As System.EventArgs) Handles
              ProdottoBindingNavigatorSaveItem.Click

        Dim list As IList = ProdottoBindingSource.List
        Dim miofile As FileStream = File.OpenWrite(
                                      "prodotti.txt")
        Dim wr As StreamWriter = New
                            StreamWriter(miofile)
        For Each prod As prodotto In list
            wr.WriteLine("{0};{1};{2}", prod.id,
                         prod.nome, prod.prezzo)
        Next
        wr.Close()
    End Sub

End Class
```

COME POSSO SAPERE SE UN GIORNO È FERIALE O FESTIVO?

UTILIZZANDO LA FUNZIONE WEEKDAY SI PUÒ RICAVARE UNA DETERMINATA DATA E SE CORRISPONDE A SABATO O DOMENICA

VISUAL BASIC.NET

FACCIAMOLO IN VISUAL BASIC

1 Sulla form trasciniamo un *DatePicker*, una label e un button

2 Clicchiamo due volte sul bottone per generare il template di gestione dell'evento onclick. Il codice di gestione è il seguente

```
Private Sub Button1_Click(ByVal sender As
        System.Object, ByVal e As System.EventArgs)
                                Handles Button1.Click
        Label1.Text = Me.checkdate(
                                DateTimePicker1.Text)
    End Sub
```

3 Poco al di sotto del codice di gestione dell'evento scriviamo il codice per il metodo checkdate

```
Function checkdate(ByVal MiaData As String)
    Select Case Weekday(MiaData)
        Case FirstDayOfWeek.Saturday
            Return "Festivo"
        Case FirstDayOfWeek.Sunday
            Return "Festivo"
        Case Else
            Return "Feriale"
    End Select
End Function
```

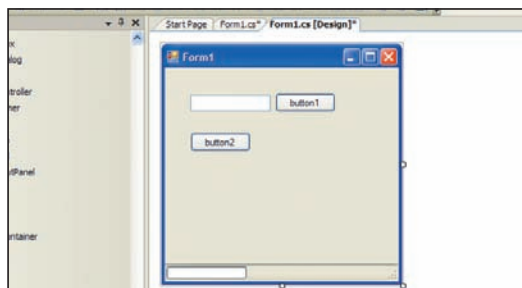
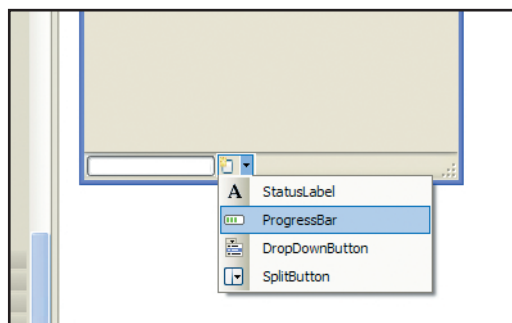
COME FUNZIONA

Alla pressione del tasto viene invocato il metodo checkdate che ritorna una stringa "Feriale" o "Festivo" che valorizza la label.
La stringa viene individuata tramite la funzione weekday. Tale funzione prende in input una data e ritorna un numero corrispondente ad un giorno della settimana. Se questo numero corrisponde a 0 o a 1 la funzione ritorna "Festivo" altrimenti "Feriale"

UNA PROGRESS BAR PER LA COPIA DEI FILE

IN MOLTI CASI È UTILE TENERE SOTTO CONTROLLO LO STATO DI UN'OPERAZIONE ATTRAVERSO UN PROGRESS BAR. VEDIAMO COME FARE

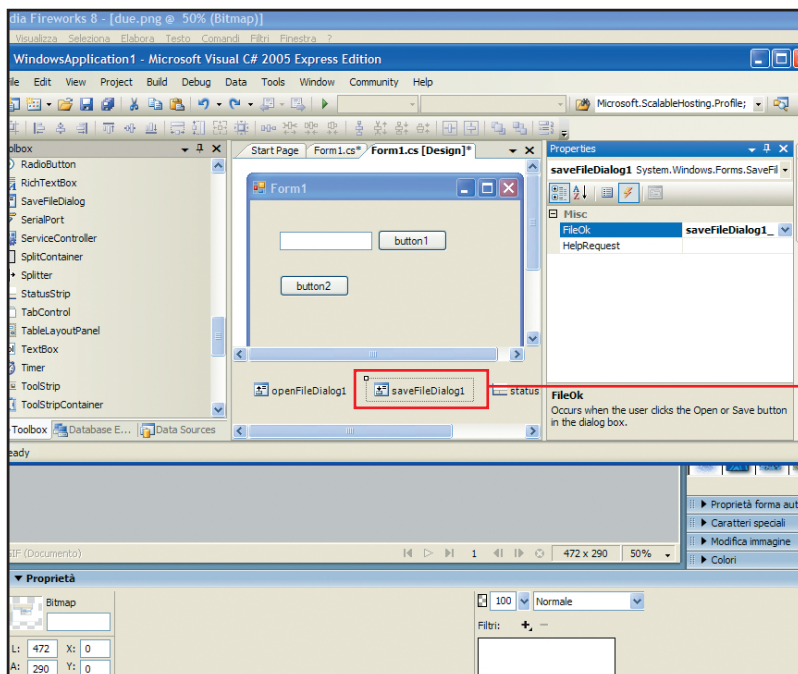
FACCIAMOLO IN C#

1 Posizioniamo sulla form una textbox, due bottoni, una statusstrip, un timer, un componente opendialog e un componente savedialog.

2 Posizioniamoci sulla statuslstrip e, agendo sullo smarttag contestuale, aggiungiamo la progress bar

3 Clicchiamo due volte sul bottone *"uno"* per generare il template del codice per l'evento *onclick*. Il codice da generare sarà il seguente

```
private void button1_Click(object sender,
                                    EventArgs e)
   {
       openFileDialog1.ShowDialog();
   }
```

4 Portiamoci sul bottone *"due"* e clicchiamo due volte per generare il template di gestione del relativo evento *click*. Il codice da inserire è il seguente

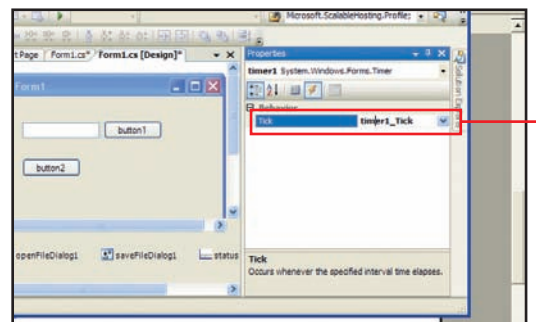
```
private void button2_Click(object sender, EventArgs e)
   {
       saveFileDialog1.FileName =
                       openFileDialog1.FileName;
       saveFileDialog1.ShowDialog();

   }
```

5 Selezioniamo il componente *SaveDialog1* e nella tabsheet degli eventi clicchiamo due volte su *"FileOk"* per fare generare il relativo template del codice

6 Il codice da inserire per la gestione dell'evento è il seguente:

```
private void saveFileDialog1_FileOk(object sender,
                                CancelEventArgs e)
{
  Thread MyNewThread = new Thread(new
                          ThreadStart(salvafile));
      MyNewThread.Start();
      MyNewThread.Join();
      timer1.Enabled = true;
      FileInfo fi = new FileInfo(textBox1.Text);
  toolStripProgressBar1.Maximum = (int)fi.Length;

}
```

7 Poco al di sotto scriviamo il codice per la gestione del metodo *salvafile*

```
private void salvafile()
 {
    if (System.IO.File.Exists(textBox1.Text))
    {
       try
       {
          System.IO.File.Copy(textBox1.Text,
                 saveFileDialog1.FileName, true);
       }
       catch
       {
          System.IO.IOException ioEx =
                   new System.IO.IOException();
          MessageBox.Show(ioEx.Message);
       }
    }
 }
```

8 Selezioniamo il component *Timer* e nella tabsheet degli eventi clicchiamo due volte sull'evento *Tick* per generare il relativo template per il codice di gestione

9 Il codice da inserire è il seguente

```
private void timer1_Tick(object sender, EventArgs e)
   {

       FileInfo fi = new FileInfo(
                   saveFileDialog1.FileName);
       if (toolStripProgressBar1.Value
                < toolStripProgressBar1.Maximum)
       {
          toolStripProgressBar1.Value =
                                (int)fi.Length;
```

```
        }
        else
        {
            toolStripProgressBar1.Value=0;
            timer1.Enabled = false;
        }

    }
```

COME FUNZIONA

Con la pressione del Bottone 1 viene mostrata la finestra di dialogo pilotata dal componente OpenDialog, che ci consente di scegliere il file da copiare. Con la pressione del Bottone 2 viene mostrata la finestra di dialogo che ci consente di salvare il file in una determinata posizione. L'unica accortezza è settare preliminarmente il nome del file da salvare con lo stesso nome scelto dalla OpenDialog in precedenza, altrimenti inizialmente ci troveremmo una stringa vuota e dovremmo digitare il nome del file a mano. Quando l'utente clicca ok sulla form di Salvataggio, si scatena un evento di tipo FileOk. In questo evento abbiamo creato un thread che avvia la procedura di copia del file tramite il metodo salvafile. Contemporaneamente viene avviato il timer che aggiorna la progress bar. Quando l'operazione di copia è terminata il timer viene disabilitato. Abbiamo preferito usare un thread per evitare operazioni di congelamento della form durante la copia di file di dimensioni generose.

FACCIAMOLO IN VISUAL BASIC

1 I componenti da copiare sulla form rimangono identici a quelli utilizzati in C#

2 Il codice di gestione relativo alla pressione del pulsante uno diventa:

```
Private Sub Button1_Click(ByVal sender As
        System.Object, ByVal e As System.EventArgs)
                                Handles Button1.Click
    OpenFileDialog1.ShowDialog()
    TextBox1.Text = OpenFileDialog1.FileName
End Sub
```

3 Il codice di gestione relativo alla pressione del pulsante due diventa

```
private void button2_Click(object sender, EventArgs e)
    {
        saveFileDialog1.FileName =
                    openFileDialog1.FileName;
        saveFileDialog1.ShowDialog();

    }
```

4 Il codice per la gestione dell'evento *FileOk* relativo che si scatena quando l'utente individua la posizione dove salvare il file diventa:

```
private void saveFileDialog1_FileOk(object sender,
                                CancelEventArgs e)
    {
        Thread MyNewThread = new
                Thread(new ThreadStart(salvafile));
        MyNewThread.Start();
        MyNewThread.Join();
        timer1.Enabled = true;
        FileInfo fi = new FileInfo(textBox1.Text);
        toolStripProgressBar1.Maximum =
                                    (int)fi.Length;

    }
```

5 La salvafile che viene gestita dal thread diventa:

```
 private void salvafile()
 {
        if (System.IO.File.Exists(textBox1.Text))
        {
            try
            {
                System.IO.File.Copy(textBox1.Text,
                    saveFileDialog1.FileName, true);
            }
            catch
            {
                System.IO.IOException ioEx = new
                        System.IO.IOException();
                MessageBox.Show(ioEx.Message);
            }
        }
    }
```

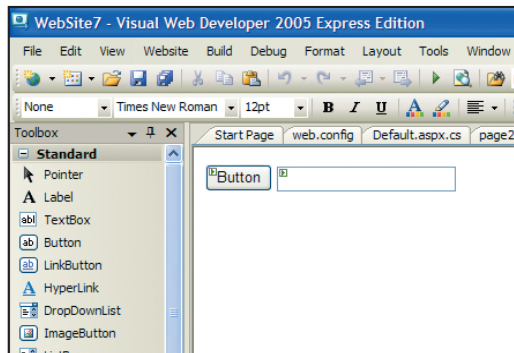
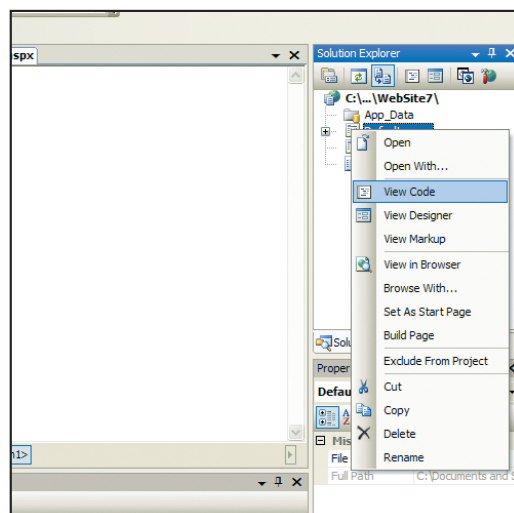
6 Infine la trasposizione per il timer è la seguente

```
private void timer1_Tick(object sender, EventArgs e)
    {

        FileInfo fi = new FileInfo(
                    saveFileDialog1.FileName);
        if (toolStripProgressBar1.Value
                < toolStripProgressBar1.Maximum)
        {
            toolStripProgressBar1.Value =
                                (int)fi.Length;
        } else
        {
            toolStripProgressBar1.Value=0;
            timer1.Enabled = false;
        }
    }
```

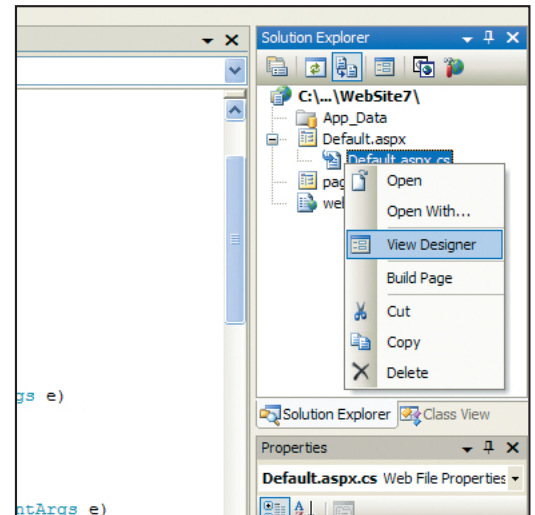
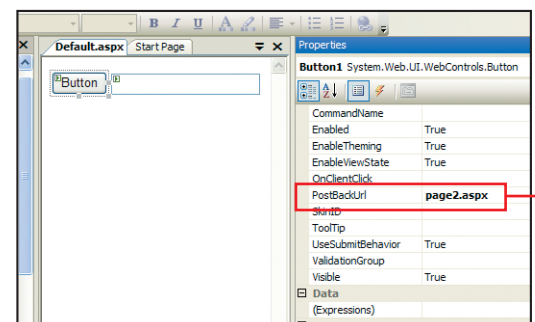
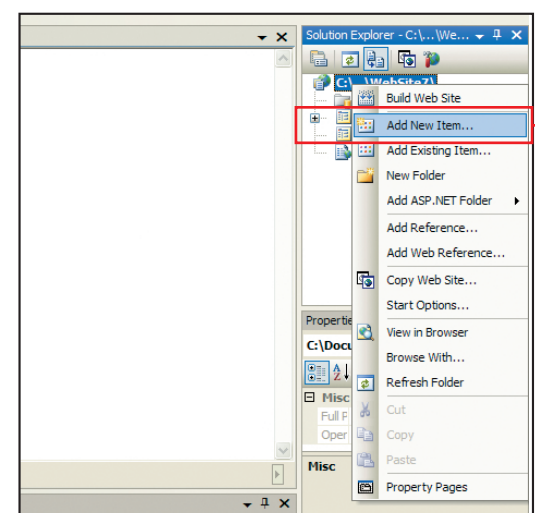
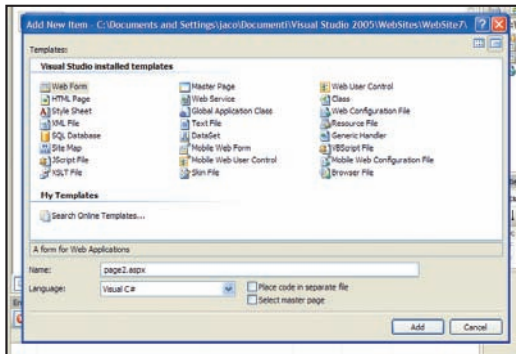
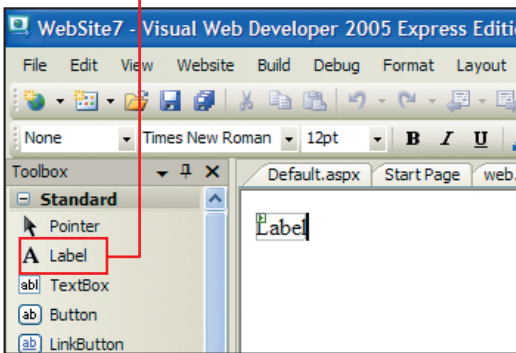
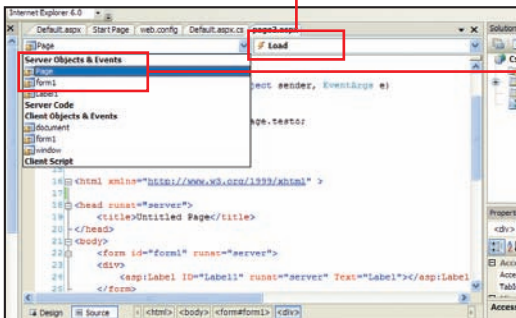
PASSAGGIO DI PARAMETRI TRA DUE PAGINE IN ASP.NET

GLI SVILUPPATORI DI APPLICAZIONI WEB UTILIZZANO PAGINE DIVERSE PER L'INSERIMENTO DEI DATI (FORM) E LA LORO ELABORAZIONE. CON ASP.NET ED IL CODE-BEHIND QUESTA CARATTERISTICA È VENUTA MENO, MA NON SIGNIFICA CHE NON SIA POSSIBILE FARLO

FACCIAMOLO IN C#

1 Trasciniamo dalla toolbox sulla pagina un bottone e un controllo di tipo TextBox

2 Passiamo in modalità codice cliccando con il tasto destro del mouse nel solution explorer sulla pagina *Default.aspx*

3 Dichiariamo una property public contenente il seguente codice

```
public string testo {
    get
    {
        return TextBox1.Text;
    }
}
```

4 Riportiamoci in modalità *Designer* cliccando con il tasto destro del mouse nel solution Explorer su *"View Designer"*

5 Selezioniamo il bottone e settiamo la sua property *"PostBackUrl"* a *page2.aspx*

6 Clicchiamo con il tasto destro del mouse nel solution explorer e scegliamo *"Add New Item"*

7 Scegliamo *"Web Form"* e assicuriamoci che il nome sia *"Page2.aspx"*

8 Nella pagina appena generata aggiungiamo una *label*

9 Spostiamoci nella modalità sorgente cliccando su *"Source"* in basso e selezionando *"Page"* nel combobox degli oggetti e *load* nel combo degli eventi associati.

10 nel template di codice generato aggiungiamo le seguente istruzioni

```
protected void Page_Load(object sender, EventArgs e)
  {
    Label1.Text = PreviousPage.testo;
  }
```

11 Spostiamoci alla prima riga della pagina e inseriamo il seguente codice:

```
<%@ PreviousPageType VirtualPath="Default.aspx" %>
```

COME FUNZIONA

In Asp2.0 è possibile settare una pagina di PostBack per un controllo, quello che abbiamo fatto per il bottone. Allo stesso tempo abbiamo dichiarato una property pubblica che restituisce il valore contenuto nel textbox. Nella pagina 2 che è anche la pagina di postback del bottone abbiamo inserito la direttiva PreviousPage indicando che la pagina chiamante sarà "Default.aspx".
A questo punto possiamo prelevare il valore contenuto nella property che abbiamo settato come public in precedenza. Siamo ovviamente costretti a usare una property public, altrimenti non sarebbe possibile ricavare il contenuto di un oggetto protetto nelle pagine seguenti.

CON LA VERSIONE 1.1 DEL FRAMEWORK

Vediamo come risolvere il problema con la versione 1.x del framework. Ipotizziamo di avere Page1.aspx, con un form contenente una textbox e un button, e Page2.aspx per l'elaborazione dei dati immessi. Nel gestore del click sul button di Page1.aspx inseriamo questo codice:

```
Private Sub Button1_Click(ByVal
    sender As System.Object, ByVal e
      As System.EventArgs) Handles
                          Button1.Click
  Server.Transfer("~/Page2.aspx")
End Sub
```

Nel Page_Load di Page2.aspx utilizziamo il contesto di esecuzione per recuperare la pagina chiamante:

```
Dim previousPage As Page1 =
   DirectCast(context.Handler, Page1)
Response.Write(
             previousPage.TextBox1.Text)
```

IN VISUAL BASIC

1 I componenti da trascinare nella *Default-.aspx*, così come nella *page2.aspx* sono identici e il procedimento di creazione rimane il medesimo. Per creare la property testo all'interno di *Default .Aspx* invece il codice è il seguente:

```
Public Property testo() As String
    Get
        Return TextBox1.Text
    End Get
    Set(ByVal value As String)

    End Set
End Property
```

2 Ricordatevi di settare la property "PostBackUrl" a page2.aspx. E in page2.aspx il codice di gestione dell'evento Load della pagina diventa:

```
  Protected Sub Page_Load(ByVal sender As Object,
                      ByVal e As System.EventArgs)
    Label1.Text = PreviousPage.testo
End Sub
```

RIMUOVERE ELEMENTI DA UN ARRAYLIST

COME POSSO SCORRERE UN ARRAY ED ELIMINARE UN ELEMENTO SE CORRISPONDE AD UN CRITERIO?

VISUAL BASIC.NET

C#

Proviamo a scorrere un ArrayList e contestualmente rimuoviamo alcuni degli oggetti in esso contenuti. Ci troveremo davanti ad una bella eccezione InvalidOperationException. Questo perché l'indice viene modificato durante la lettura. Provate a trascinare su una form un componente di tipo listbox e un bottone. Riempite la listbox con dei valori casuali, ad esempio "uno", "due", "tre" e associate all'evento onclick sul bottone il seguente codice

```
Private Sub Button1_Click(ByVal sender As
        System.Object, ByVal e As System.EventArgs)
                                Handles Button1.Click
    Dim index As Integer = 0
    For Each item As Object In ListBox1.Items
        If item = "uno" Then
            ListBox1.Items.RemoveAt(index)
        End If
        index = index + 1
    Next
End Sub
```

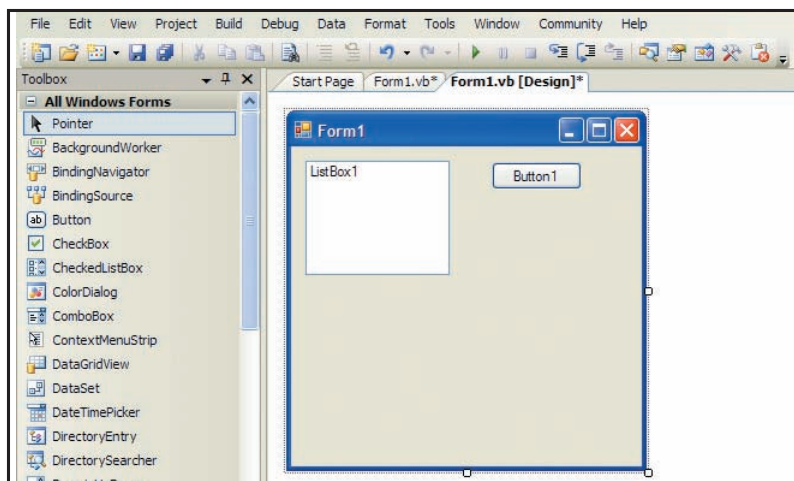
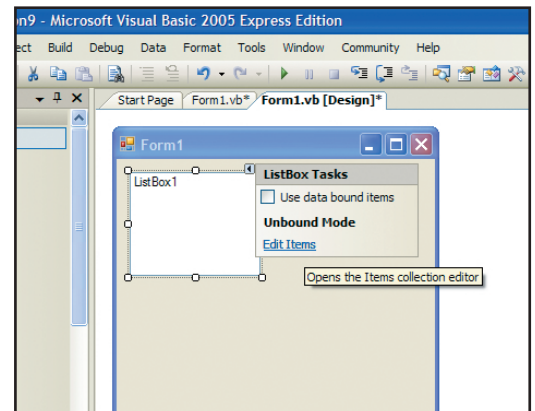
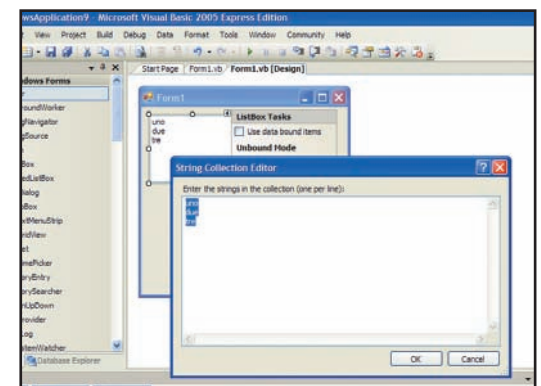
I NAMESPACE DA USARE

In c# è necessario aggiungere all'inizio del codice la seguente riga di codice

```
using System
.Collections;
```

Noterete che alla pressione del bottone si ottienete proprio l'errore di cui sopra. Vediamo come risolverlo.

FACCIAMOLO IN VISUAL BASIC

1 Trasciniamo su una Form dalla toolbox un componente ListBox e un bottone

2 Dallo Smart Tag associato alla ListBox selezioniamo "Edit Items"

3 Nella Dialog Box che compare digitate uno per riga i valori *"uno", "due", "tre"* senza le virgolette

4 Cliccate due volte sul *"Button1"* per far generare il template di gestione dell'evento *"OnClick"* che diventa

```
Private Sub Button1_Click(ByVal sender As
        System.Object, ByVal e As System.EventArgs)
                                Handles Button1.Click
    Dim itemsToDelete As New ArrayList
    For Each item As Object In ListBox1.Items
        If item = "uno" Then
            itemsToDelete.Add(
                    ListBox1.Items.IndexOf(item))
        End If
    Next

    Dim i, index As Integer

    For i = (itemsToDelete.Count - 1) To 0 Step -1
        index = CType(itemsToDelete(i), Integer)
        ListBox1.Items.RemoveAt(index)
    Next
```

COME FUNZIONA

Creiamo un array d'appoggio chiamato itemsToDelete, cicliamo la collection items della listbox e se il valore di un item corrisponde a un criterio, aggiungiamo il suo indice all'array d'appoggio. A questo punto in un'operazione successiva cicliamo l'array ottenuto e rimuoviamo dalla listbox tutti i valori il cui indice è contenuto in itemsToDelete

FACCIAMOLO IN C#

1 I passi da uno a tre rimangono identici, il codice di gestione dell'evento *OnClick* sul bottone diventa invece

```
private void button1_Click(object sender, EventArgs e)
      {  ArrayList itemsToDelete = new ArrayList();
         foreach (object item in listBox1.Items) {
            if (item.ToString() == "uno")  {
               itemsToDelete.Add(
                    listBox1.Items.IndexOf(item));}
            }
      for (int i = itemsToDelete.Count - 1; i >= 0; i--)
      {  int index = (int)itemsToDelete[i];
         listBox1.Items.RemoveAt(index);
      }
```

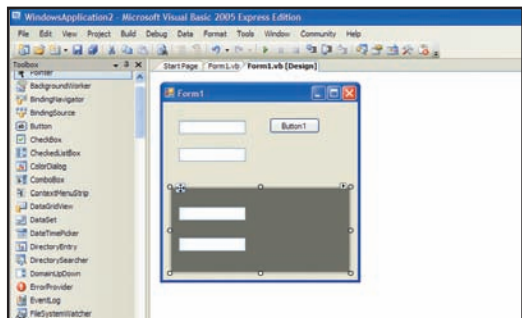
COME "CANCELLARE" IL CONTENUTO DI UNA FORM

IN ALCUNI CASI È UTILE POSIZIONARE UN PULSANTE "RESET" SU UNA FORM, PERCHÉ ALLA SUA PRESSIONE TUTTI I CAMPI DELLA FORM SIANO RIPORTATI AD UN VALORE INIZIALE.

VISUAL BASIC.NET

C#

FACCIAMOLO IN VISUAL BASIC

1 Trasciniamo sulla form due textbox e un Panel, al cui interno inseriremo altre due textbox, infine aggiungiamo un bottone

2 clicchiamo due volte sul bottone per generare il template per la gestione dell'evento OnClick. Il codice da inserire è il seguente:

```
Private Sub Button1_Click(ByVal sender As
     System.Object, ByVal e As System.EventArgs)
                             Handles Button1.Click
    clearform(Me)
End Sub
```

3 Poco più in basso digitiamo il codice del metodo clearform, come segue:

```
Sub clearform(ByVal ctrl As Control)
    If TypeOf ctrl Is TextBox Then
        ctrl.Text = ""
    End If
    For Each child As Control In ctrl.Controls
```

```
            clearform(child)
        Next
    End Sub
```

COME FUNZIONA

Alla pressione del bottone viene richiamato il metodo clearform a cui viene passato come parametro l'oggetto form stesso. Il metodo in questione controlla se l'oggetto passato come parametro è un textbox e in tal caso lo svuota. Se non lo è cicla su tutti gli oggetti contenuti in quello base, e per ogni oggetto richiama di nuovo se stesso assando come parametro l'oggetto corrente nella lista. Il metodo è ricorsivo.
Quando il ciclo finisce esce dal metodo.

FACCIAMOLO IN C#

1 I passi per disegnare la form rimangono sostanzialmente identici. Il codice di gestione dell'evento OnClick relativo al bottone diventa

```
clearform(this);
```

2 Il metodo clearform, da scrivere poco al di sotto di quello generato in precedenza è :

```
public void clearform(Control ctrl)
      {
         if (ctrl is TextBox)
         { ctrl.Text=""; }
         foreach (Control child in ctrl.Controls) {
            clearform(child); }
      }
```

VISUAL STUDIO 2005 EXPRESS È GRATIS?

LA MAGGIOR PARTE DEGLI ESEMPI PROPOSTI IN QUESTO NUMERO DI IOPROGRAMMO FA USO DI VISUAL STUDIO 2005 EXPRESS EDITION. CERCHIAMO DI CAPIRE DI PIÙ SUL PERCHÉ CONVIENE UTILIZZARE QUESTO NUOVO STRUMENTO

IN COSA DIFFERISCE LA VERSIONE EXPRESS DALLA VERSIONE STANDARD

Si tratta di una versione esplicitamente pensata per gli hobbysti o le piccole software house. Questo non significa che sia limitata in una qualche maniera, ma semplicemente che non dispone di alcuni strumenti avanzati quali ad esempio quelli per lo sviluppo im team presenti nella versione teamsystem, tuttavia contiene tutti gli strumenti delle versioni "standard" e risponde in modo efficace alla maggior parte delle esigenze di un normale programmatore.

È GRATIS?

Si per un anno. Microsoft ha proposto un prezzo iniziale di 49$ per poi effettuare invece una promozione che consente il download gratuito del prodotto presso il sito http://msdn.microsoft.com/vstudio/express. La promozione durerà fino al 7 Novembre 2006. Dopodicché, gli utenti che avranno scaricato e installato la versione Express, potranno comunque continuare a utilizzarla senza limitazioni.
In questo periodo è in promozione con la stessa logica anche SQL Server 2005 express edition, tuttavia non è meglio specificata la durata della promozione per quanto riguara questo prodotto.

LE VERSIONI EXPRESS GODONO DEL SUPPORTO DI MICROSOFT?

No, non c'è un supporto ufficiale. Ma MS ha creato un forum all'indirizzo *https://forums.microsoft.com/msdn/ShowForum.aspx?ForumID=24* a cui si può fare riferimento *Express Compatibility and Versioning*

POSSO USARE LA VERSIONE EXPRESS PER FINI COMMERCIALI?

Assolutamente sì. Non ci sono limitazioni di licenza per le applicazioni costruite con la versione Express. Potete tranquillamente creare le vostre applicazioni e distribuirle

POSSO AVERE SITI WEB CHE USANO ASP.NET 1.1 E ASP.NET 2.0 SULLA STESSA MACCHINA?

Sì, possono essere usati insieme e il controllo di versione è abbastanza semplice. Semplicemente una tabsheet in IIS stabilisce se un sito debba girare con ASP .NET 1.1 o ASP.NET 2.0

POSSO SVILUPPARE APPLICAZIONI PER IL FRAMEWORK 1.1 CON LE VERSIONI EXPRESS?

No, questa versione supporta solo il framework 2.0. Le novità proposte nel nuovo framework e la potenza dell'ambiente giustificano ampiamente il passaggio alla nuova versione.
Noi di ioProgrammo suggeriamo di passare velocemente a Visual Studio 2005. Il tool è studiato in modo che la produttività ne ricava sicuramente benefici elevati.

VISUAL STUDIO EXPRESS È COMPATIBILE CON LE ALTRE VERSIONI DI VISUAL STUDIO?

Tipicamente sì, sia con le altre versioni della stessa serie sia con quelle precedenti. Tuttavia in caso di problemi di conversione da una versione precedente esiste una lista di compatibilità all'indirizzo *http://msdn.microsoft.com/netframework/programming/breakingchanges/default.aspx*.

DEVO INSTALLARE IIS PER CREARE APPLICAZIONI PER IL WEB?

No, Visual Web Developer Edition contiene un Web Server Integrato, che svincola completamente dall'installazione di IIS.
Questo risolve anche tutti i problemi che avevano coloro che utilizzavano Windows XP Home Edition.

IL CODICE SVILUPPATO CON VISUAL BASIC 6.0 FUNZIONA IN VISUAL BASIC EXPRESS?

Visual Basic Express contiene un Wizard che facilita la migrazione, in ogni caso molto probabilmente sarà necessario fare ancora qualche variazione manuale.
Per i pochi che ancora si ostinano a sviluppare con Visual Basic 6.0, il passaggio alla nuova versione del framework è ormai diventato una necessità.

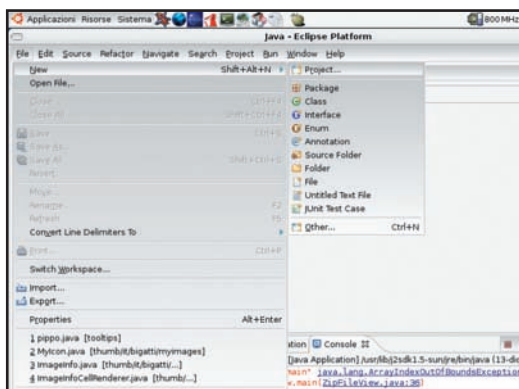
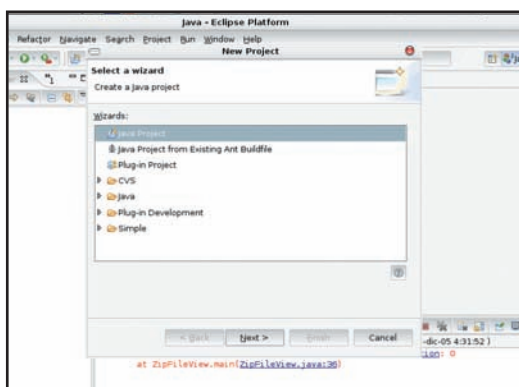
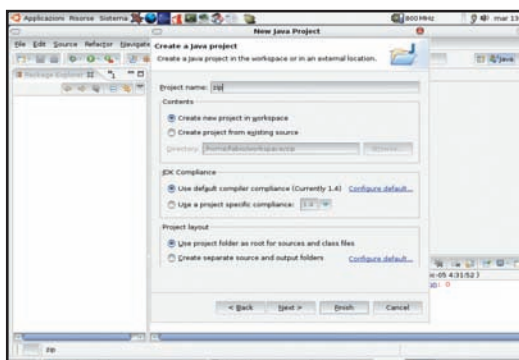
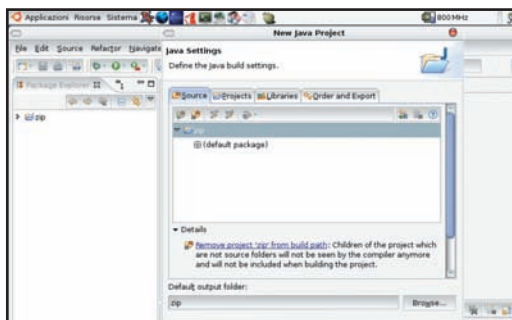
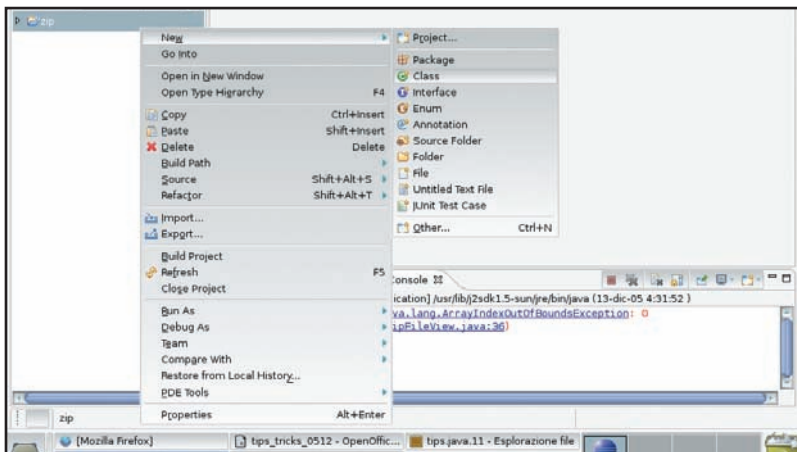
CHE COSA È SQL SERVER 2005 EXPRESS

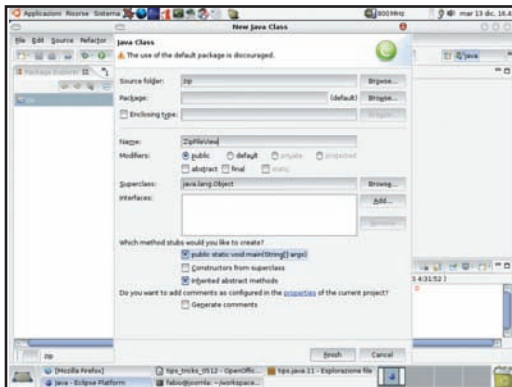
Sostanzialmente è una versione ridotta di SQL Server 2005, potete immaginarlo come il naturale erede di MSDE. Offe tutte comodità tipiche di SQL Server 2005, tra cui il fatto di essere basato su file, estremamente portabile dunque, veloce, con grande attenzione alle sue performance e dotazioni di sicurezza.
Le limitazioni riguardano il fatto che è possibile usare una sola CPU, 4GB di spazio su disco per database, e un GB di Ram.
Inoltre non vengono inclusi i componenti di analisi e di reporting.

COME ELENCARE IL CONTENUTO DI UNO ZIP FILE

IN QUESTO TIP DESCRIVEREMO COME ELENCARE LE ENTRIES CONTENUTE ALL'INTERNO DI UNO ZIP FILE. DI SEGUITO VIENE ILLUSTRATA UNA SEMPLICE CLASSE IL CUI METODO MAIN RICEVE COME PARAMETRO IL PERCORSO DI UNO ZIP FILE E NE STAMPA LE INFORMAZIONI A VIDEO

FACCIAMOLO CON ECLIPSE

1 Iniziamo con il creare un nuovo progetto cliccando su “file/new/project”

2 Fra quelli messi a disposizione di Eclipse scegliamo “Java Project”

3 Scegliamo come nome del progetto “zip” e lasciamo intatte le altre opzioni così come ci vengono proposte

4 Anche in questa schermata lasciamo intatte le opzioni di default e procediamo

5 Facciamo click con il tasto destro del mouse sul progetto appena creato e dal menu contestuale scegliamo “New/Class”

6 Chiamiamo la classe “ZipFileView” e assicuriamoci che sia “spuntato” il checkbox “Public Static void main(string[] args)

ECLIPSE

7 Nel codice che viene creato automaticamente aggiungiamo all'inizio

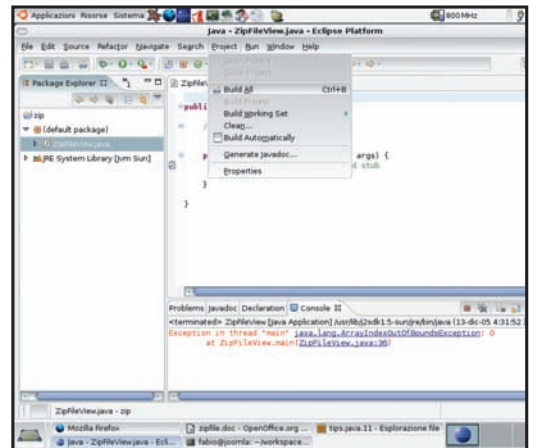
```
import java.util.Enumeration;
import java.util.zip.ZipEntry;
import java.util.zip.ZipFile;
import java.io.IOException;
```

8 Modifichiamo il metodo main anch'esso automaticamente creato come segue

```
public static void main(String[] args)
{
    ZipFileView.execute(args[0]);

}
```

9 Poco più sotto aggiungiamo il codice che implementa il metodo execute

```
public static void execute(String file)
{
  try
  {
    // apre il file ...
    ZipFile zipfile = new ZipFile(file);
    // e ne ottiene tutte le entry
    ZipEntry zipEntry;
    Enumeration entries = zipfile.entries();
    while (entries.hasMoreElements())
    {
      // per ogni entry ...
      zipEntry = (ZipEntry)
                       entries.nextElement();
      // ne stampa le informazioni (nome,
               dimensione reale e dimensione
                                compressa)
      System.out.print("Entry: " +
                          zipEntry.getName());
      System.out.print("- Size: " +
                           zipEntry.getSize());
      System.out.println("- Compressed Size:
                " + zipEntry.getCompressedSize());
    }
  }
  catch (IOException e)
  {
    e.printStackTrace();
  }
}
```

10 Portiamoci su “Project/build all” e clicchiamo per compilare il nostro progetto

11 Portiamoci nella directory che contiene il nostro progetto compilato e da una console digitiamo java ZipFileView nomefile.zip dove nomefile.zip è il nome del file zip compreso di percorso di cui vogliamo ottenere le informazioni. Se tutto è andato a buon fine otterremo l'elenco dei file contenuti nello zip con le relative dimensioni.

COME FUNZIONA?

Il metodo main richiama il metodo execute, che al suo interno lo cicla per ottenere le informazioni desiderate. Le due classi base che contengono i metodi per realizzare i nostri scopi sono zipfile e zipentry.

COME POSSO CREARE UNA COLLEZIONE DI OGGETTI?

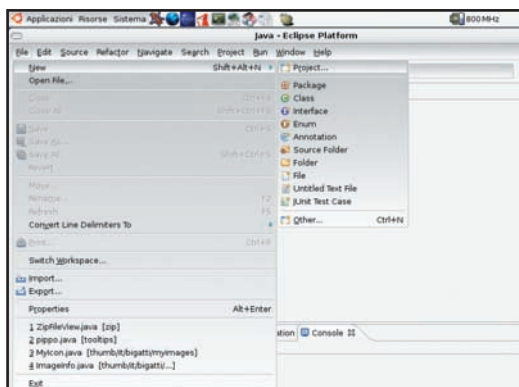
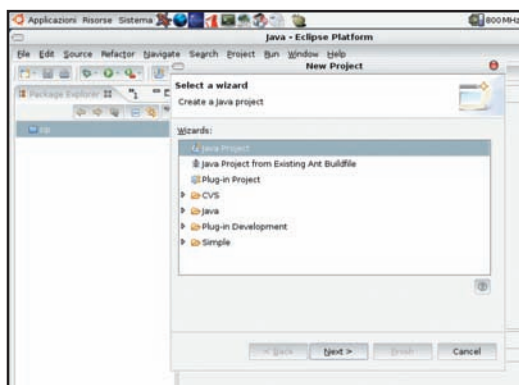
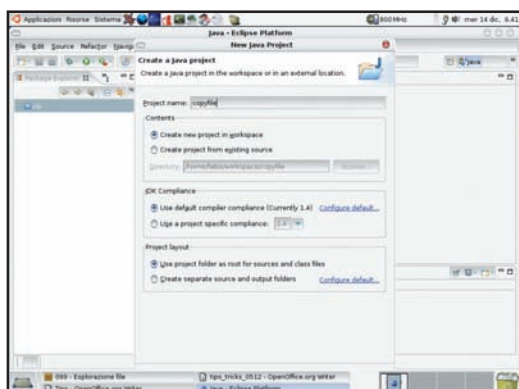
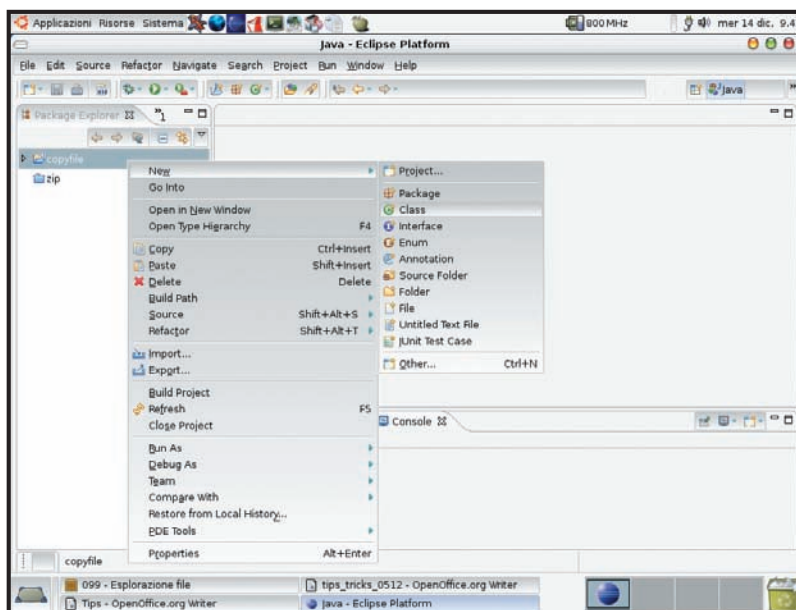
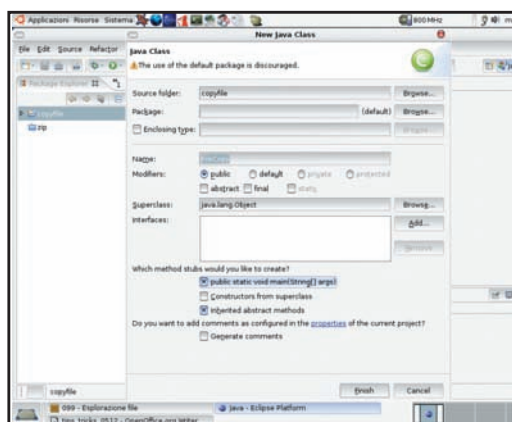
Un metodo utile che consente di strutturare i dati in modo efficace. L'implementazione è abbastanza semplice.
In questo caso utilizziamo una collezione implementata da un ArrayList.
Volendo basta cambiare classe e lasciare il resto uguale (ad es si può usare LinkedList). Successiva scansione della lista con un iteratore.

```
Collection macchine = new ArrayList();
macchine.add(“Punto”);
macchine.add(“Brera”);
macchine.add(“Z4”);
Iterator iterMacchine = macchine.iterator();
while(iterMacchine.hasNext())
{
    String macchina = (String)
                     iterMacchine.next();
    System.out.println(“Macchina nella
                     lista: ” + macchina);
}
```

COME POSSO COPIARE UN FILE CON NIO?

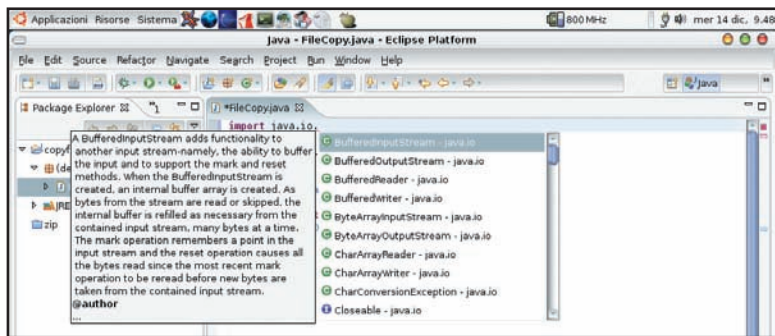
LE API NIO RAPPRESENTANO UN MODO SEMPLICE E VELOCE PER GESTIRE I FILE IN AMBITO JAVA. VEDIAMO COME USARLE

FACCIAMOLO CON ECLIPSE

1 Iniziamo da File/New/Project per creare un nuovo progetto Java

2 Selezioniamo Java Project e andiamo avanti

3 Chiamiamo il progetto “copyfile” e lasciamo invariate le altre opzioni proposte. Clicchiamo su next e su finish nelle altre due schermate proposte senza effettuare nessuna variazione

4 Aggiungiamo una nuova classe cliccando con il bottone destro del mouse sul nome del progetto e selezionando “New/Class” nel menu contestuale

5 Chiamiamo la classe “FileCopy” e assicuriamoci che sia spuntato il checkbox che ci consente di creare in automatico il metodo main

PERCHÉ SU WINDOWS E SU LINUX LA VELOCITÀ È DIVERSA?

Le API NIO, introdotte con la versione 1.4 della J2SE. Ricordiamo che, a parità di risorse hardware, le prestazioni possono cambiare radicalmente se eseguite su sistemi operativi differenti, in quanto la tecnologia NIO sfrutta le system call del sistema operativo sottostante.

LE FACILITY DI ECLIPSE

Digitando le prime righe dell'import a esempio *import java.* e poi utilizzando la combinazione di tasti *"CTRL+TAB"* si accede ad un menu a tendina contenente un elenco di possibili import che completano le righe digitate. Una comodità non da poco.

6 Aggiungiamo all'inizio del codice i seguenti import

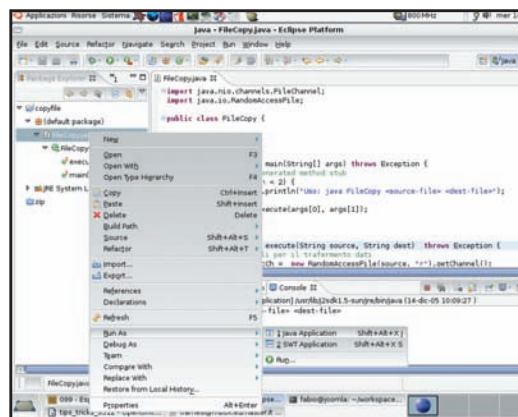
```
import java.io.RandomAccessFile;
import java.nio.channels.FileChannel;
```

7 Il metodo main della classe conterrà le seguenti righe:

```
public static void main(String[] args)
                    throws Exception
  {  if (args.length < 2)
       { System.out.println("Uso: java FileCopy
                <source-file> <dest-file>"); }
     else {
       FileCopy.execute(args[0], args[1]); }
}
```

8 Poco più in basso implementiamo il metodo execute come segue:

```
public static void execute(String source, String dest)
  throws Exception
  {  // apre i canali per il trafermento dati
     FileChannel srcCh =
     new RandomAccessFile(
                    source, "r").getChannel();
     FileChannel destCh = new RandomAccessFile(
                    dest, "rw").getChannel();
     // trasferisce tutti i byte dal canale di lettura a
                    quello di scrittura
     srcCh.transferTo(0, srcCh.size(), destCh);
     // chiude i canali
     srcCh.close();
     destCh.close();
  }
```

9 Portiamoci con il tasto destro del mouse sul file che contiene la classe e compiliamola scegliendo "Run As/Java Application"

10 Portiamoci in una console, nel percorso che contiene FileCopy.class e digitiamo java FileCopy nomefilesorgente filedestinazione. Se tutto è andato a buon fine il file verrà copiato nella locazione corrente

COME FUNZIONA

Il metodo main rappresenta l'entry point dell'applicazione. Non fa altro che prendere i due parametri dalla linea di comando e passarli al metodo execute. Se i parametri non sono due o non corrispondono a due file viene generata un'eccezione. Il metodo execute apre due canali di comunicazione e fisicamente copia i file utilizzando NIO e il metodo transferTo.

COME POSSO SCOMPORRE UNA FRASE?

In alcuni casi è utile analizzare il contenuto di una frase estraendo le parole che la compongono, vediamo come. Un primo metodo è utilizzare un oggetto di tipo StringTokenizer come segue

```
StringTokenizer st = new StringTokenizer(
                    "Ciao a tutti");
  while (st.hasMoreTokens()) {
     System.out.println(st.nextToken());}
```

Se invece si vogliono estrarre le parole separate da un carattere speciale è possibile utilizzare il metodo split come segue:

```
String stringa = new String("04-07-78");
String[] elementi;
elementi = stringa.split("-");
```

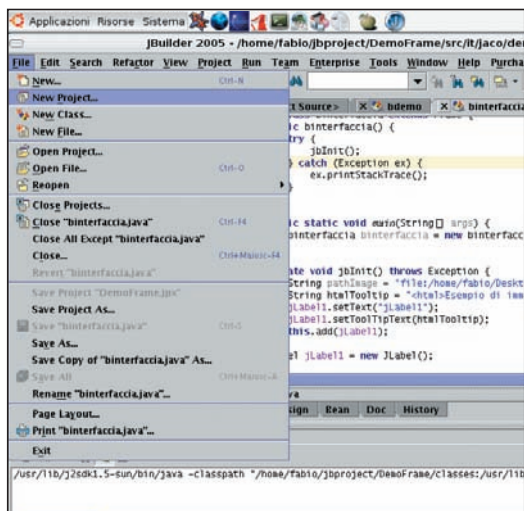
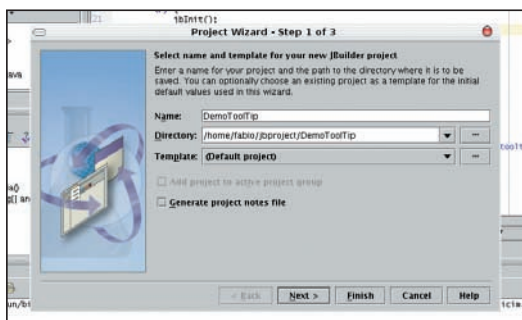
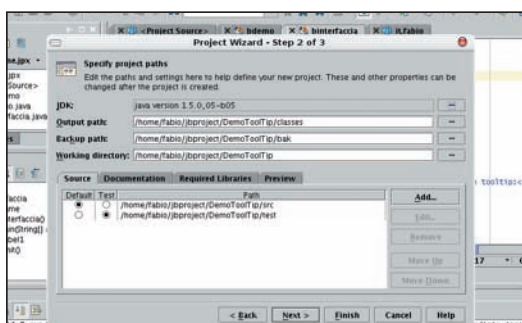
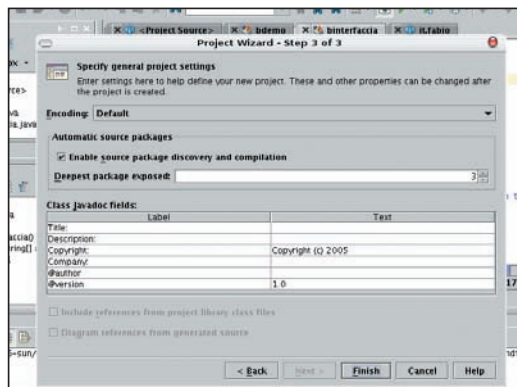
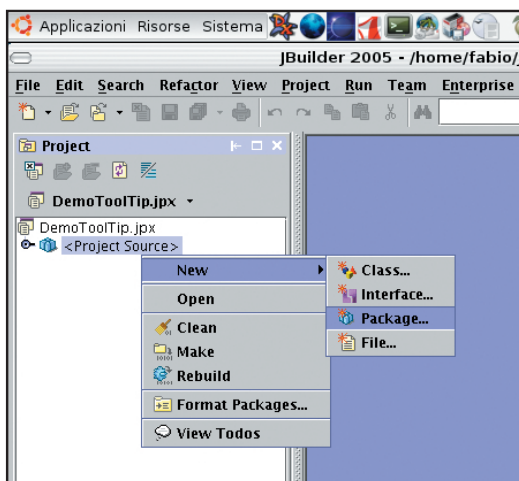
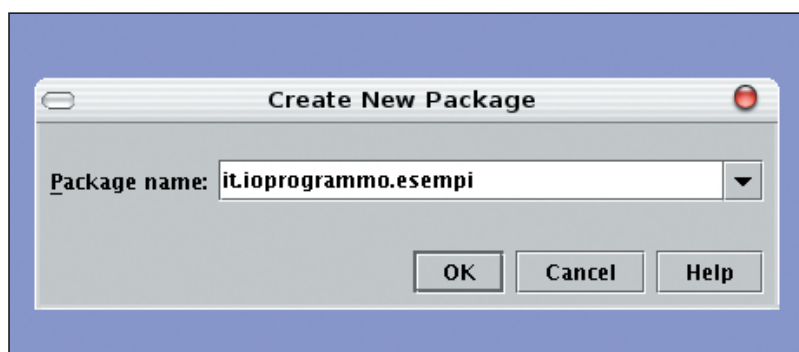
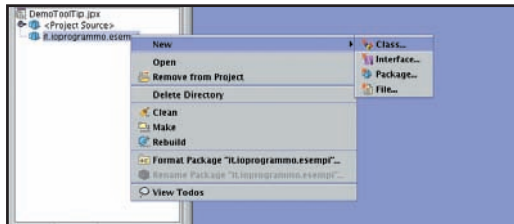
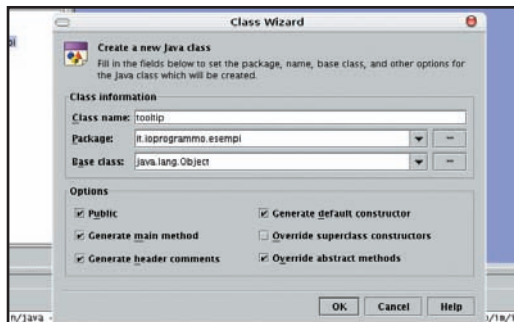
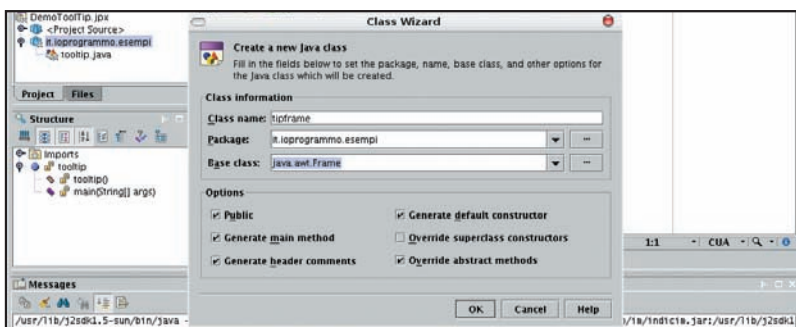
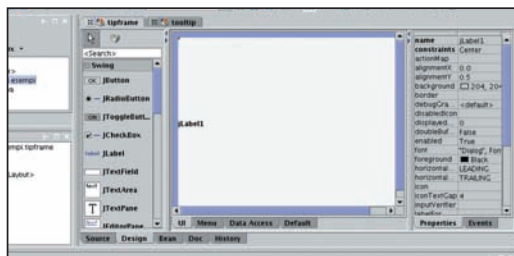
restituisce i valori:
elementi[0] = 04; elementi[1] = 07; elementi[2] = 78;

COME VISUALIZZARE UN'IMMAGINE ALL'INTERNO DI UN TOOLTIP

UN METODO SEMPLICE CHE ARRICCHISCE DI INFORMAZIONI LA NOSTRA APPLICAZIONE, QUANDO L'UTENTE SI SOFFERMERÀ SU UNA LABEL, VERRÀ VISUALIZZATO UN TOOLTIP COMPRENSIVO DI UN'IMMAGINE

JAVA BUILDER

FACCIAMOLO IN JBUILDER

1 Iniziamo da file/new project

2 Scegliamo come nome del progetto *"DemoTool-Tip"*

3 Controlliamo che tutti i percorsi siano conformi alle nostre esigenze e clicchiamo su "Next"

4 Lasciamo le impostazioni di default e andiamo avanti

5 Facciamo click con il tasto destro del mouse sul nome del progetto e scegliamo "New/Package"

6 Il nome del package sarà "it.ioprogrammo .esempi"

7 Clicchiamo con il tasto destro del mouse sul package appena creato e aggiungiamo una classe agendo sul menu “New/Class”

8 La nuova classe si chiamerà tooltip

9 Aggiungiamo ancora una classe dal nome tipframe, questa volta assicuriamoci che la classe base sia java.awt.frame

10 Passiamo in modalità design e aggiungiamo una label al Frame principale

11 Passiamo in modalità sorgente e modifichiamo il metodo jbInit come segue

```
private void jbInit() throws Exception {
    String pathImage = "file:
        /home/fabio/Desktop/test.jpg";
    String htmlTooltip = "<html>Esempio di
        immagine" + "all'interno di un tooltip:
        <img src=? + pathImage + ?></html>";
    jLabel1.setText("jLabel1");
    jLabel1.setToolTipText(htmlTooltip);;
    this.add(jLabel1, java.awt.BorderLayout.CENTER);
}
```

12 Spostiamoci nella classe tooltip e modifichiamo il costruttore della classe come segue

```
public tooltip() {createUi(); }
```

13 Poco più sotto aggiungiamo il codice di gestione del metodo createUI

```
private void createUi() {
    boolean framepack = false;
    tipframe frame = new tipframe   ();
    if (framepack) {
        frame.pack();
    } else { frame.validate();}
    Dimension screenSize =
        Toolkit.getDefaultToolkit().getScreenSize();
  Dimension frameSize = frame.getSize();
  if (frameSize.height > screenSize.height) {
    frameSize.height = screenSize.height; }
  if (frameSize.width > screenSize.width) {
    frameSize.width = screenSize.width; }
  frame.setLocation((screenSize.width
        - frameSize.width) / 2, (screenSize.height
            - frameSize.height) / 2);
  frame.setVisible(true);
}
```

14 Infine modifichiamo il metodo main

```
public static void main(String[] args) {
    try {
        UIManager.setLookAndFeel(UIManager
         .getCrossPlatformLookAndFeelClassName());}
    catch(Exception e) {
        e.printStackTrace(); }
    tooltip tooltip = new tooltip(); }
```

COME FUNZIONA

Prima di tutto grande merito va a Jbuilder usandolo lo scoprirete molto leggero, facile da usare e molto utile nella costruzione rapida di interfacce java. Per quanto riguarda il nostro esempio la classe tooltip non fa altro che invocare la classe la classe tipframe all'interno della quale viene impostata la costruzione dell'interfaccia grafica. Da notare che per la visualizzazione dell'immagine all'intero del tooltip utilizziamo semplice Html.

COME POSSO RIEMPIRE UNA LISTA CON SOLO ALCUNI DATI DI UN DATABASE?

CONCEDIAMO ALL'UTENTE DI SELEZIONARE I DATI TRAMITE UN CHECKBOX E POI INSERIRLI IN UNA LISTA TRAMITE UN BOTTONE

ASP.NET

C#

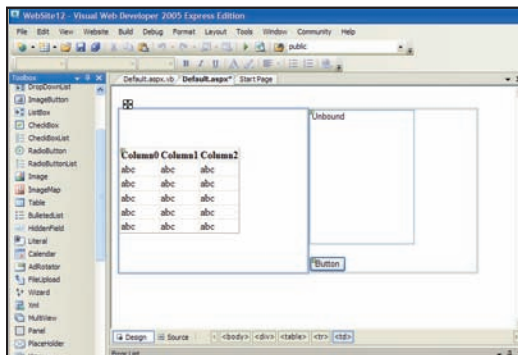
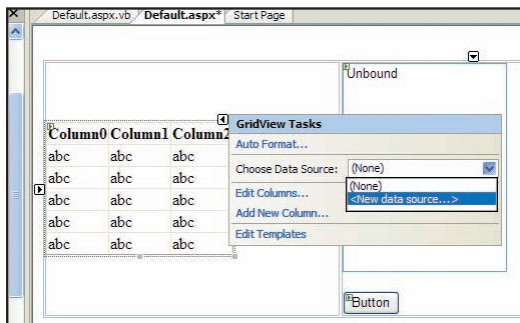
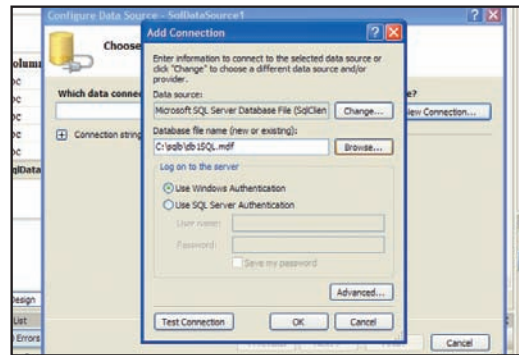
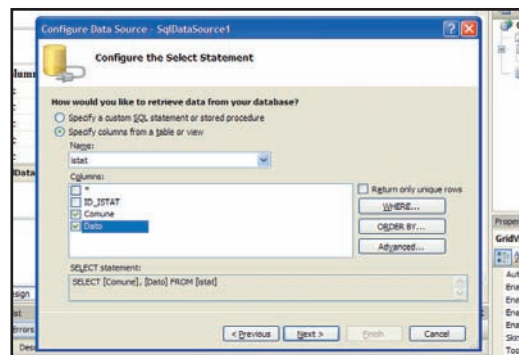
PHP

La flessibilità di XML consiste nel poter utilizzare file composti ad esempio come segue:

```
<?xml version="1.0" encoding="UTF-8"?>
<item>dato</item>
```

qualunque programma in grado di leggere il file XML in questione sarà in grado di interpretare quanto in esso contenuto. Proprio perché le regole di XML sono standard e certe non bisogna reinventare i parser ogni volta.

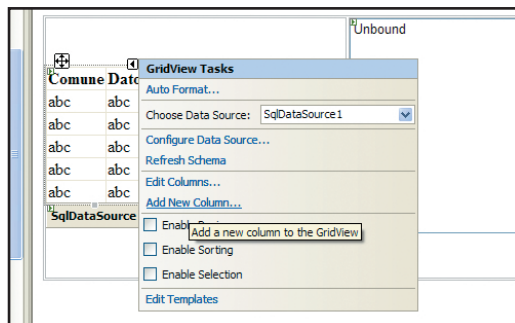
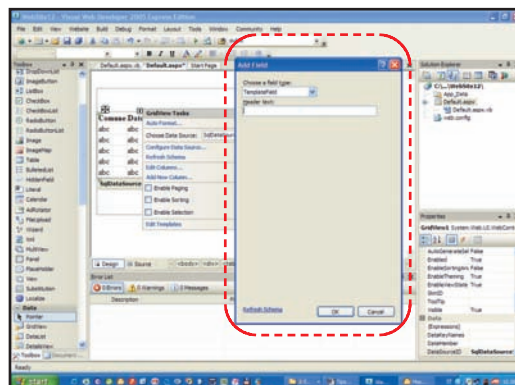
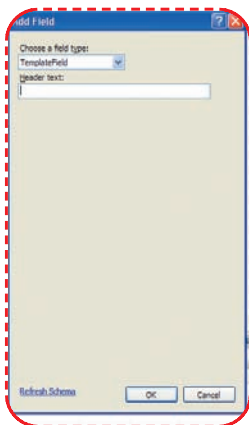
FACCIAMOLO IN ASP.NET

1 Iniziamo trascinando dalla toolbox sulla pagina default.aspx una gridview, una listbox e un bottone

2 Dallo Smart Tag associato alla gridview scegliamo "New Data Source"

3 Nella schermata successiva scegliamo DataBase e poi Ok. Nella successiva ancora scegliamo "New Connection". Infine nell'ultima schermata selezioniamo la fonte dati che ci interessa, nel nostro caso abbiamo scelto "Microsoft Sql Server file DataBase File". Tramite il tasto browse scegliamo dunque il database che contiene il file.

4 Clicchiamo su Next fino a quando non compare la schermata che ci consente di scegliere la tabella da associare alla gridview, nel nostro caso è una tabella che contiene i comuni d'Italia. Scegliamo anche quali campi visualizzare nella gridview. Infine clicchiamo sempre next fino al termine della procedura

LA PAROLA CHIAVE DIRECTCAST

Viene utilizzata per effettuare un'operazione di conversione, esattamente come la parola chiave Ctype. Deve essere richiamata passandogli due valori. Il primo valore punta all'oggetto da convertire, mentre il secondo determina il tipo di destinazione. La differenza fra CType e Directcast consiste nella loro capacità di effettuare una conversione quando a runtime il tipo di destinazione è diverso da quello di partenza. Facciamo un esempio:

```
Dim Num As Object = 2.45
Dim I As Integer = CType(Num,
        Integer) ' Questa operazione
        restituisce esito positivo
Dim J As Integer = DirectCast(Num,
        Integer) ' Questa operazione
        restituisce esito negativo
```

La seconda operazione fallisce perché a runtime il valore di Num non è un intero. Se si è certi che l'operazione DirectCast è possibile, è conveniente utilizzare questa funzione piuttosto che la Ctype, in quanto leggermente più veloce.

5 Dallo Smart Tag associato alla gridview selezioniamo "Add New Column"

6 Selezioniamo come campo "TemplateField" possiamo anche lasciare in bianco il valore di header text

7 Dallo Smart Tag associato questa volta selezioniamo "Edit Templates"

8 Dallo Smart Tag associato al template questa volta selezioniamo ItemTemplate e trasciniamo nello spazio relativo un componente di tipo checkbox. Quando abbiamo terminato clicchiamo su "End Template Ending"

9 Clicchiamo due volte sul bottone per fare generare il codice di gestione legato all'evento OnClick, che sarà:

```
Protected Sub Button1_Click(ByVal sender As
                    Object, ByVal e As System.EventArgs)
                                    Handles Button1.Click
    For Each row As GridViewRow In
                                          GridView1.Rows
        Dim check As CheckBox =
                          DirectCast(row.FindControl(
                                 "CheckBox1"), CheckBox)
        If check.Checked Then
            Dim item As New ListItem(
                                     row.Cells(1).Text)
            If Not Me.ListBox1.Items.Contains(item)
                                                   Then
                Me.ListBox1.Items.Add(item)
            End If
        End If
    Next
End Sub
```

COME FUNZIONA

Al click dell'utente sul bottone viene effettuato un ciclo sugli elementi della GridView. In ciascuna riga viene cercato il controllo CheckBox1 e viene associato a un oggetto di tipo checkbox chiamato check. Se il controllo check risulta selezionato viene aggiunto alla listbox. Prima viene controllato con un if che non vi fosse stato aggiunto in precedenza.

EFFICACIA DI PHP+JAVASCRIPT

Si noti che in queste particolari condizioni, a parità di difficoltà e di tempo nella stesura del codice, l'implementazione ASP.NET appare molto più rapida ed efficiente. Con un paio di clic di mouse in più in ASP.NET avremmo potuto implementare la paginazione dei dati e inoltre il mantenimento della sessione avrebbe garantito che anche paginando la listbox non si sarebbe svuotata. Viceversa scrivere le funzioni di paginazione e di mantenimento della sessione in PHP avrebbe comportato uno sforzo molto maggiore. In questo particolari casi, la soluzione Microsoft appare conveniente, almeno nella velocità con cui è possibile stendere il codice di base.

FACCIAMOLO IN ASP.NET CON C#

1 I passi dall'uno al nove rimangono identici. Il codice di gestione del bottone diventa invece

```
protected void Button1_Click(object sender,
                                         EventArgs e)
```

```
  {  foreach (GridViewRow Row in GridView1.Rows) {
          CheckBox check = (CheckBox)
                         (Row.FindControl("CheckBox1"));
          if (check.Checked) {
              ListItem item = new
                              ListItem(Row.Cells[1].Text);
              if (! ListBox1.Items.Contains(item)) {
                   ListBox1.Items.Add(item); }  }   }
  }
```

FACCIAMOLO IN PHP CON JAVASCRIPT

1 Iniziamo con la connessione al database, e con una query che recupera i dati che ci servono

```
<?php
   $connessione = mysql_connect("localhost",
                            "nomeutente", "password")
       or die("Connessione non riuscita: " .
                                        mysql_error());
   print "Connesso con successo";
   mysql_select_db("mailservice") or die("Selezione
                          del database non riuscita");
   /* Esecuzione di una query SQL */
   $query = "SELECT * FROM istat limit 10";
   $risultato = mysql_query($query) or die("Query
                           fallita: " . mysql_error() );
?>
```

2 Cicliamo i dati ottenuti e stampiamo la form che li conterrà

```
<form name=elencodati id=elencodati >
<?
   /* Stampa dei risultati in HTML */
   echo "<table>\n";
   while ($linea = mysql_fetch_array($risultato,
                                    MYSQL_ASSOC)) {
       echo "\t<tr>\n";
       foreach ($linea as $valore_colonna) {
           echo "\t\t<td>$valore_colonna</td>\n";
       }
       echo "\t\t<td><input type=checkbox
                            id=checkbox1[] value=
                    "'.$linea['Comune']."'> </td>\n";
       echo "\t</tr>\n";
   }
   print "</table>\n";
?>
```

3 Aggiungiamo alla stessa form un bottone e una listbox

```
<select name="select2" id="select2" size="15">
  </select>
   <input type="Button" onClick=
              "javascript:addtolist()" id="Aggiungi"
                                  value="Aggiungi">
</form>
```

4 Liberiamo la memoria

```
<?
  /* Liberazione delle risorse del risultato */
   mysql_free_result($risultato);
   /* Chiusura della connessione */
   mysql_close($connessione);
?>
```

5 Scriviamo la funzione JavaScript che si scatenerà alla pressione del bottone

```
function addtolist(){
   var objForm = document.forms[0];
   var elSel = document.getElementById('select2');
   var el = document.getElementsByName(
                                    'checkbox1[]')
   for (f=0;elSel.length;f++) {
            elSel.options[f]=null;
    }
    for(i=0;i<el.length;i++){
    if(el[i].checked){
    elSel.options[elSel.length]=new
                         Option(el[i].value, "");
           f++;
     }
   }
}
```

È POSSIBILE USARE FUNZIONI .NET CON PHP?

Èsiste un modulo di PHP che consente di accedere ad Assemly.NET, ovvero *dotnet_load*. Ma funziona?
Si tratta di un modulo molto sperimentale, da utilizzare con cautela a causa della sua affidabilità ancora tutta da testare. Prima di tutto è necessario registrare l'assembly nella GAC, per farlo potete usare l'utility REGASM, e di seguito la GACUTIL. Una volta fatto questo è abbastanza semplice utilizzare la vostra classe da PHP utilizzando una sintassi del tipo:

```
$obj = new COM ("NameSpace.Class");
$value = $obj ->MyFunction($parameter)
```

CONVERTIRE I DATI DA MYSQL AD ACCESS

IN QUALCHE CASO PUÒ ESSERE UTILE COME SOFTWARE INTERMEDIO VERSO ALTRI FORMATI O PER FARE DELLE ELABORAZIONI SPECIFICHE SU SISTEMI MICROSOFT

FACCIAMOLO DA PHPMYADMIN

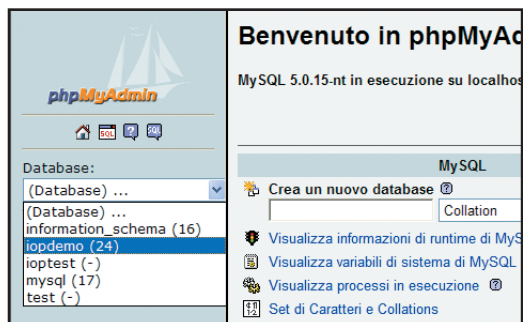
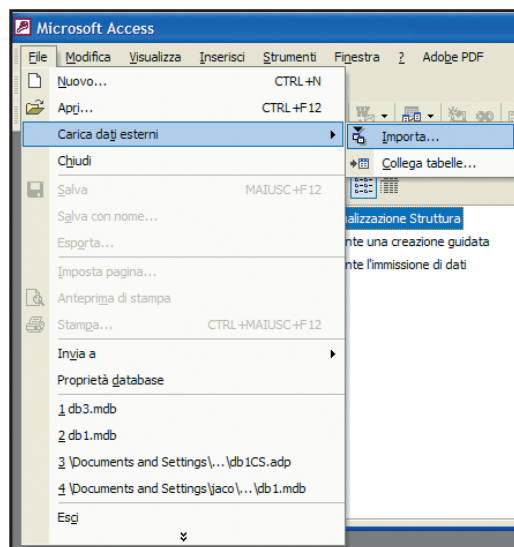
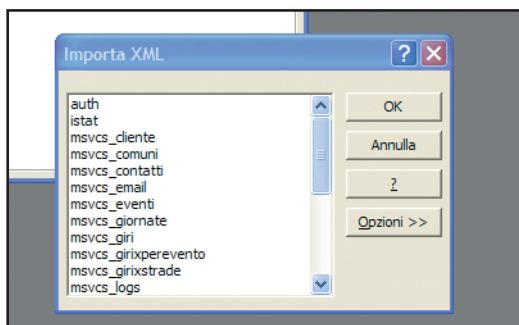
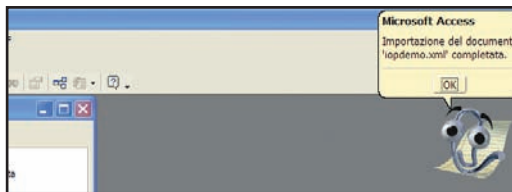
1 Puntiamo il browser sul percorso di installazione di PhpMyAdmin, nel nostro caso *http://localhost/phpmyadmin/* e selezioniamo il database che ci interessa esportare

2 Nella tabsheet superiore clicchiamo su "Esporta"

Tabella	Azione	Record	Tipo	Collation	Dimensione	In eccesso
auth		46	MyISAM	latin1_swedish_ci	6,2 KB	-
istat		8.103	MyISAM	latin1_swedish_ci	353,8 KB	-
msvcs_cliente		9	MyISAM	latin1_swedish_ci	2,9 KB	-
msvcs_comuni		8.102	MyISAM	latin1_swedish_ci	812,8 KB	-
msvcs_contatti		16	MyISAM	latin1_swedish_ci	4,7 KB	-
msvcs_email		18	MyISAM	latin1_swedish_ci	2,9 KB	-
msvcs_eventi		5	MyISAM	latin1_swedish_ci	2,4 KB	-
msvcs_giornate		76	MyISAM	latin1_swedish_ci	10,8 KB	-
msvcs_giri		3	MyISAM	latin1_swedish_ci	2,1 KB	-
msvcs_girixperevento		1	MyISAM	latin1_swedish_ci	1,0 KB	-
msvcs_girixstrade		7	MyISAM	latin1_swedish_ci	1,2 KB	-
msvcs_logs		13	MyISAM	latin1_swedish_ci	2,6 KB	-
msvcs_menu		7	MyISAM	latin1_swedish_ci	2,3 KB	-

3 Selezioniamo "formato XML" e spuntiamo il checkbox "Salva con nome", infine clicchiamo su esegui

4 Da access creiamo un nuovo database vuoto

5 Clicchiamo su "File/Carica Dati Esterni/Importa" e selezioniamo il file XML salvato in precedenza. Avendo cura di selezionare "tipo file documento xml" dalla finestra che appare per individuare il percorso del file

6 Selezioniamo le tabelle da importare, oppure clicchiamo direttamente su Ok e diamo il via alla fase di importazione

7 Se tutto va a buon fine è sufficiente ciccare su “Ok” sull’assistant per avere a disposizione il nuovo database in formato Access

COME POSSO EVIDENZIARE I CAMPI DI UNA FORM IN MODALITÀ DI EDITING?

LO SCRIPT CHE VI PROPONIAMO CAMBIA DINAMICAMENTE IL COLORE DI UNA LABEL ASSOCIATA AD UNA TEXTBOX QUANDO UN UTENTE SI POSIZIONE SU DI ESSA

JAVASCRIPT

FACCIAMOLO IN JAVASCRIPT

1 Scriviamo la funzione evidenzia, che prende come parametro un id, e la funzione unselect che riport tutto allo stato originale

```
<script language="JavaScript" type="text/javascript">
//<![CDATA[
function evidenzia(id)
{  Elemento = document.getElementById(id);
   Elemento.style.color = "#FF0000"; }
function unselect(id)
{  Elemento = document.getElementById(id);
   Elemento.style.color = "#0000FF"; }
 //]]
</script>
```

2 Componiamo uno stile che in un secondo momento varieremo dinamicamente

```
<style type="text/css">
.campo{
font-family:Courier;
color:#0000FF;
font-weight:bold; }
</style>
</head>
<body>
```

3 Disegniamo la form che ci interessa

```
<form name="form1" id="form1" method="GET">
  <table width="600" border="0" align="center"
                    cellpadding="0" cellspacing="5">
    <tr>
      <td align="right">
         <div class="campo" id="Campo1L">
                    Campo1:</div>
         </td>
         <td>
           <input name="Campo1" type="text" id=
                "Campo1" size="30" onfocus=
               "evidenzia('Campo1L')" onblur=
                   "unselect('Campo1L')" />
         </td>
    </tr>
    <tr>
      <td align="right">
           <div  class="campo" id=
              "Campo2L">Campo2:</div></td>
             <td>
               <input name="Campo2" type=
                  "text" id="Campo2" size="30"
                 onclick="evidenzia('Campo2L')"
               onblur="unselect('Campo2L')"  />
             </td>
    </tr>
    <tr>
      <td> </td>
      <td align= "right">
        <input name="Fine" type="hidden" id="Fine"
                                    value=""/>
           <input type="submit" name="Submit"
                              value="Invia" />
      </td> </tr></table></form>
```

COME FUNZIONA?

Quando l’utente entra nella textbox si scatena un evento onclick che richiama la funzione evidenzia, che varia il foglio di stile associato al campo. Quando l’utente lascia la textbox si scatena un evento di tipo OnBlur a cui è associata la funzione OnSelect che riporta tutto allo stato precedente.

COME POSSO SAPERE QUANTI GIORNI MANCANO AD UN EVENTO?

IN PHP POSSIAMO SFRUTTARE LA FUNZIONE MKTIME, CHE NASCE PER RESTITUIRE LA UNIX TIMESTAMP, OVVERO IL NUMERO DI SECONDI TRASCORSI DAL 1 GENNAIO 1970 ALLA DATA INDICATA

PHP

FACCIAMOLO IN PHP

1 Creiamo un nuovo file miadata.php al cui interno definiamo la nuova classe miadata

```
class miadata
{
   public function dayto($start, $to) {
     $day_start=strtotime(date("d/m/y",strtotime(
                                              $start)));
     $day_to=strtotime(date("d/m/y",strtotime($to)));
     $intervallo=floor(($day_to-$day_start)/86400);
     return $intervallo;
   }
}
?>
```

2 Creiamo il file index.php includendo il riferimento alla classe appena creata

```
<?
include_once("miadata.php");
$miadata=new miadata();
print $miadata->dayto("10/10/05","10/11/06");
?>
```

COME FUNZIONA

La classe riceve una data in formato italiano. La converte in formato inglese tramite la funzione data che prende in input il timestamp della data ottenuto con strtotime e restituisce una stringa nel formato corretto. Infine viene di nuovo applicata la conversione in timestamp utilizzando di nuovo strtotime. Le due date così ottenute vengono utilizzate per ottenere il numero di secondi di distanza fra di loro. Per ottenere il numero di giorni dividiamo per 86400, ovvero 60*60*24.

COME POSSO IMPLEMENTARE IL PATTERN SINGLETON IN PHP?

QUESTO PATTERN FA SI CHE ESISTA UNA SOLA ISTANZA DI UNA CLASSE PER QUALUNQUE SUO OGGETTO VENGA CREATO. VEDIAMO COME FUNZIONA

PHP

FACCIAMOLO IN PHP

1 Creiamo un nuovo file singleton.php all'interno del quale inseriremo il codice di gestione della classe:

```
<?php
class Single
{
   private static $instance;
   private static $numero;
   private function __construct($num)
   {  $this->numero=$num;  }
   public static function singleton($num)
   {
       if (!isset(self::$instance)) {
           $c = __CLASS__;
           self::$instance = new $c($num); }
       return self::$instance; }
   public function getnumber()
   { return $this->numero;  }
   public function __clone()
   {  trigger_error('Non è possibile clonare
                              l\'oggetto.', E_USER_ERROR); }
}
?>
```

2 Creiamo un file index.php che farà uso della classe

```
<?php
include_once("singleton.php");
// L'istruzione seguente genererebbe un errore
// Il costruttore infatti è private
//$test = new Single();
$test = Single::Singleton(7);
$testn = Single::Singleton(8);
echo $test->getnumber();
echo $testn->getnumber();

// Questa istruzione genera un errore
// intercettato dal nostro trigger
// $test_clone = clone($test);

?>
```

COME FUNZIONA?

Qualunque parametro voi passiate ad ogni nuovo oggetto vedrete sempre restituirvi il valore legato al primo oggetto creato. Ogni nuovo oggetto infatti non viene fisicamente "istanziato" ma il metodo "singleton" restituisce un riferimento alla prima istanza creata. Questo vi consente di creare più oggetti che condividono le stesse proprietà.

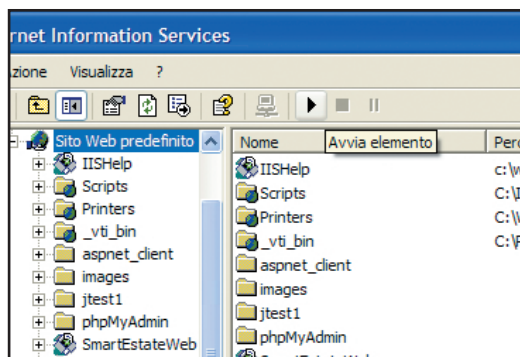
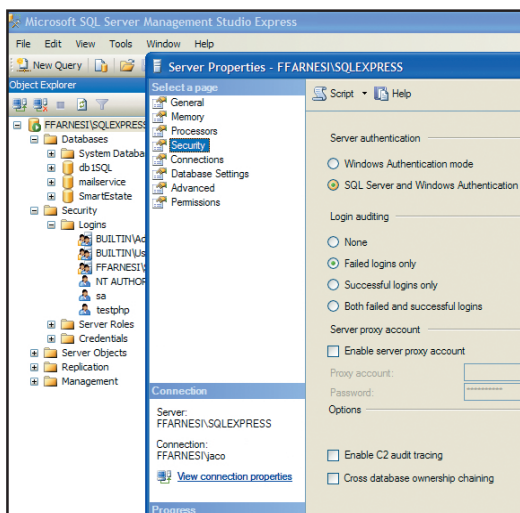
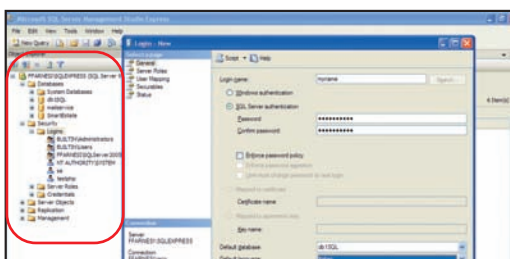
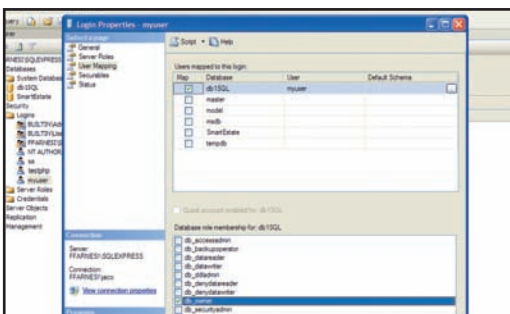
CONNETTERSI A SQL SERVER 2005 EXPRESS DA PHP

L'ARRIVO DELLA NUOVA VERSIONE DEL SERVER DI MICROSOFT APRE MOLTE NUOVE POSSIBILITÀ. VEDIAMO COME SFRUTTARLE DA PHP

FACCIAMOLO IN PHP

1 Nel file php.ini individuate l'estensione *php_mssql.dll* ed eliminate il commento il punto e virgola che la precede

```
extension= php_mssql.dll
```

2 Riavviate il server di IIS tramite la la management console

3 In SQL Server assicuratevi che sia attiva l'autenticazione "Mixed Mode"

4 A questo punto potete creare l'utente SQL Server

5 Assegnategli i permessi per accedere al database che ritenete opportuno

6 Create il file .php con il codice di prova che consente di verificare che tutto funzioni

```
<?php
$con = mssql_connect ("localhost \SQLEXPRESS",
                      "myuser", "mypassword");
mssql_select_db ("db1SQL", $con);
$sql= "SELECT * FROM table";
$rs= mssql_query ($sql, $con);
echo "The field number one is: ";
echo mssql_result ($rs, 0, 0);
mssql_close ($con);
?>
```

PHP

AUTENTICARE CON L'ISTANZA

Perché l'autenticazione avvenga in maniera corretta, dovete passare come parametro della funzione *mssql_ connect* anche il nome dell'istanza del DB di SQL Server, nel nostro caso *SQLEXPRESS*.

DOVE SCARICARE LA CONSOLE

Di default SQL Server nella versione express non viene installato con una console di amministrazione. Dovete scaricare la SQL Server Management Studio Express seguendo il link all'indirizzo http://msdn.microsoft.com/vstudio/express/sql/default.aspx

ESEGUIRE QUERY INNESTATE

L'IDEA È QUELLA DI PRELEVARE I DATI CHE CI SERVONO DAL RISULTATO DI UNA QUERY, GRAZIE AD UN'ALTRA QUERY. VEDIAMO COME

Supponiamo di volere estrarre da una tabella "Comuni" tutti i comuni che non fanno già parte di un'altra tabella, ad esempio la tabella dei Comuni serviti dai mezzi di un'azienda di trasporto.

FACCIAMOLO DA LINEA DI COMANDO

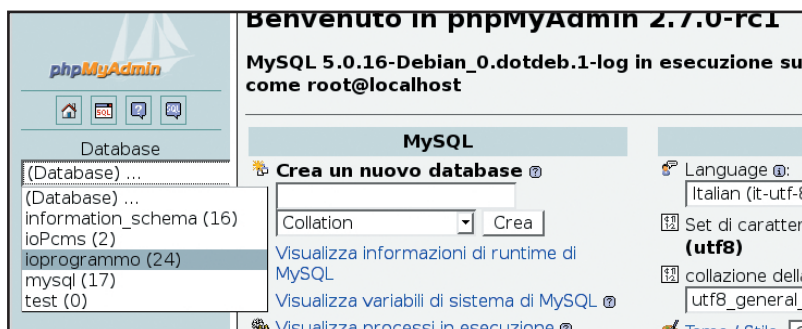
1 Connettiamoci al database mysql con i comandi

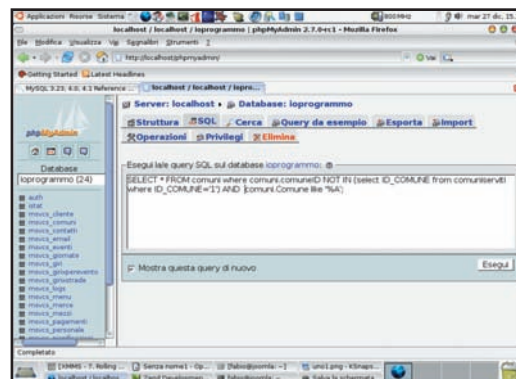
```
mysql -unomeutente -ppassword nomedatabase
```

2 Digitiamo la query

```
SELECT * FROM comuni where msvcs_
        comuni.comuneID NOT IN (select ID_COMUNE
            from comuniserviti where ID_COMUNE='1')
                AND  msvcs_comuni.Comune like '%A';
```

FACCIAMOLO DA PHPMYADMIN

1 Puntiamo il browser sull'indirizzo in cui abbiamo installato PHPMyAdmin ad esempio *http://localhost/PHPMyAdmin* e selezioniamo dal menu a sinistra il database a cui connettersi

2 Dalla schermata che compare selezioniamo "SQL"

3 Scriviamo la query che ci serve direttamente nella maschera che compare e clicchiamo su "Esegui"

FACCIAMOLO DA PHP

1 Connettiamoci al database tramite le istruzioni

```
<?php
  $connessione = mysql_connect("localhost",
                                  "root", "atp0h21")
        or die("Connessione non riuscita: " .
                                       mysql_error());
mysql_select_db('ioprogrammo', $connessione);
```

2 Inseriamo la nostra subquery in una stringa come segue

```
$query="SELECT * FROM comuni where
        comuni.comuneID NOT IN (select ID_COMUNE
            from comuniserviti where ID_COMUNE='1')
                AND  msvcs_comuni.Comune like '%A'";
```

3 Eseguiamo la query

```
$result = mysql_query($query,$connessione);
```

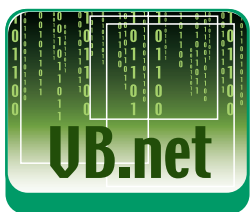
4 E recuperiamo il risultato

```
  while ($columnRow = mysql_fetch_array($result,
                                    MYSQL_ASSOC))
  {  print "<pre>";
       print_r($columnRow);
       print "</pre>";
  }
```

COME FUNZIONA?

Molto semplicemente vengono restituiti i comuni che appartengo alla tabella *comuni* e tali che non appartengano alla tabella *comuniservizi.*

Visual Basic.NET e Il file system

Non c'è applicazione che non faccia accesso in qualche modo ai file per salvare i dati, per gestire la configurazione e per scrivere l'output. Vediamo quali sono le classi per gestire al meglio file e directory

La gestione degli elementi del file system, in VB .NET 2003, è garantita dal namespace *System.IO*. Il namespace *System.IO* contiene una libreria di classi che offrono proprietà, metodi ed eventi per creare, copiare, spostare ed eliminare file e directory.
Le classi maggiormente utilizzate per la gestione di file e directory sono:

- La classe *Directory* che fornisce metodi condivisi per la gestione delle directory.
- La classe *File* che fornisce metodi condivisi per la gestione dei file.
- La classe *DirectoryInfo* che rappresenta una directory specifica e, come la classe Directory, permette di gestire cartelle (fornisce solo metodi di istanza).
- La classe *FileInfo* che rappresenta un file specifico e, come la classe File, permette di gestire file (fornisce solo metodi di istanza)
- La classe *Path* che fornisce metodi condivisi per la gestione delle informazioni sul percorso.

In tutti gli esempi di codice, per evitare di scrivere ogni volta il nome completo del namespace, diamo per scontato l'inserimento della seguente istruzione Imports:

```
Imports System.IO
```

Conoscenze richieste
Sistema operativo: Windows 2000/XP. Visual Basic .NET 2003
Software
Impegno
Tempo di realizzazione

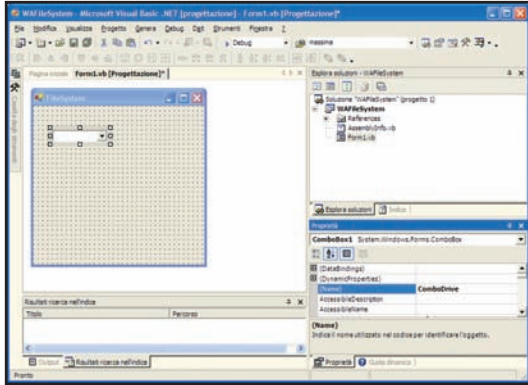
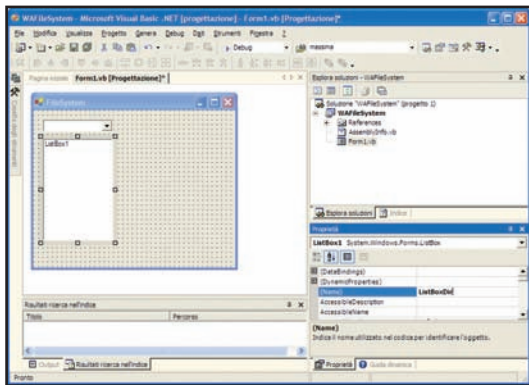
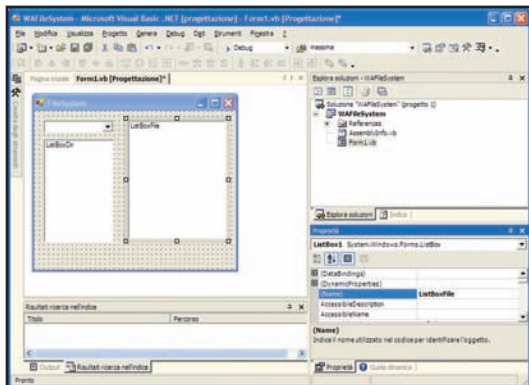
LA CLASSE DIRECTORY

La gestione delle directory è garantita dalla classe *Directory*. La classe *Directory*, permette la creazione, modifica e cancellazione delle directory e fornisce, inoltre, alcuni metodi per gestire le unità logiche del sistema (drive). I metodi della classe *Directory* sono metodi statici pertanto non è necessario creare un'istanza (la classe non fornisce un costruttore), di norma i più utilizzati sono:

- *CreateDirectory(percorso)*. Permette di creare la directory indicata nell'argomento *percorso*. *Se percorso*, contiene delle directory intermedie che non esistono, allora vengono create anch'esse.
- *Delete(percorso)*. Permette di eliminare una directory ed il relativo contenuto
- *Exists(percorso)*. Permette di stabilire se il percorso, indicato come argomento, esiste sul disco, in questo caso restituisce il valore *True*.
- *GetDirectories(percorso)*. Permette di recuperare tutte le sottodirectory contenute nella directory corrente, restituendole in una matrice di stringhe.
- *GetLogicalDrives*. Permette di recuperare i nomi delle unità logiche presenti nel computer restituendoli in una matrice di stringhe.
- *GetFiles(percorso)*. Permette di recuperare tutti i file contenuti in una directory, eventualmente filtrati secondo un criterio specifico, restituendoli in una matrice di stringhe.
- *GetParent(percorso)*. Permette di recuperare la directory padre del percorso specificato
- *Move(percorsoSorgente, percorsoDestinazione)*. Permette di spostare un oggetto *DirectoryInfo*, ed il suo contenuto, in un nuovo percorso
- *SetCurrentDirectory* e *GetCurrentDirectory*. Permettono di impostare o recuperare la directory corrente.

IN PRATICA

Mettiamo subito in azione la classe *Directory*, ed i suoi metodi, creando un nuovo progetto *Windows Applications* che ci permetta di navigare nel file system. Sulla finestra *Form1* disegniamo l'interfaccia utente:

1 Selezioniamo un controllo *ComboBox* dalla casella degli strumenti, e disegniamolo sulla form. Dalla finestra delle proprietà cambiamo il nome in *ComboDrive*. In *ComboDrive* visualizzeremo le unità logiche (drive) presenti nel sistema.

2 Selezioniamo un controllo *ListBox* dalla casella degli strumenti, e disegniamolo sulla form. Dalla finestra delle proprietà cambiamo il nome in *ListBoxDir*. In *ListBoxDir* visualizzeremo le directory presenti nell'unità logica selezionata in *ComboDrive*.

3 Selezioniamo un controllo *ListBox* dalla casella degli strumenti, e disegniamolo sulla form. Dalla finestra delle proprietà cambiamo il nome in *ListBoxFile*. In *ListBoxFile* visualizzeremo i file contenuti nella directory selezionata in *ListBoxDir*.

Le unità logiche dovranno essere mostrate appena viene visualizzata la finestra iniziale, pertanto il codice necessario dovrà essere scritto nell'evento Load della form:

```
Private Sub Form1_Load(ByVal sender As
        System.Object, ByVal e As System.EventArgs)
                                Handles MyBase.Load

End Sub
```

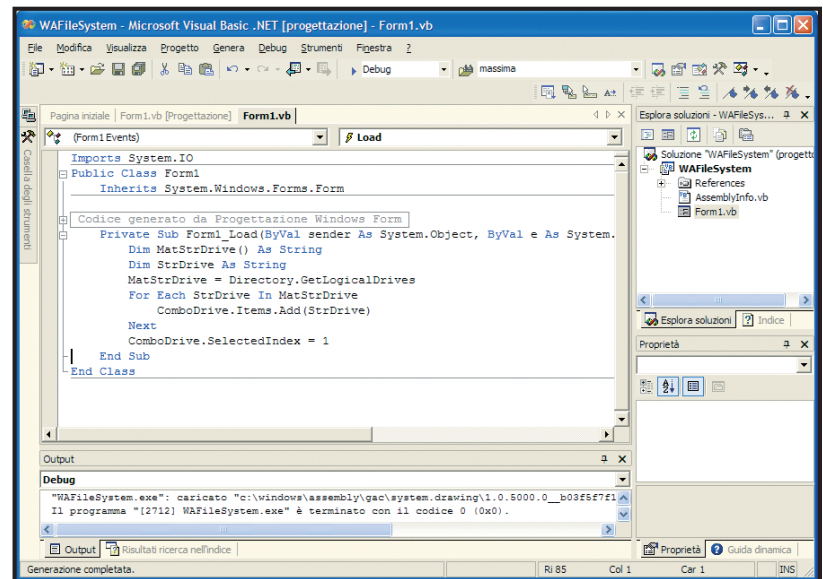



Fig. 1: ***In figura il codice***

Definiamo la variabile che dovrà contenere la matrice di stringhe delle unità logiche presenti nel computer, e la variabile che dovrà contenere il drive corrente da mostrare nel ComboBox.

```
Dim MatStrDrive() As String
Dim StrDrive As String
```

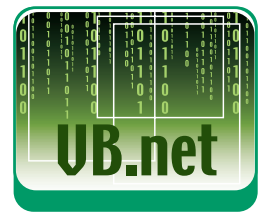
Utilizziamo il metodo GetLogicalDrives per valorizzare la matrice dei drive.

```
MatStrDrive = Directory.GetLogicalDrives
```

Con un ciclo For Each .. Next cicliamo sulla matrice dei drive ed aggiungiamo ogni elemento della matrice in ComboDrive:

```
For Each StrDrive In MatStrDrive
    ComboDrive.Items.Add(StrDrive)
Next
```

Infine mostriamo l'unita C:\ che dovrebbe avere indice uno (la prima dovrebbe essere A:\).

```
ComboDrive.SelectedIndex = 1
```

Se volessimo conoscere gli attributi di una directory non sarebbe possibile utilizzare la classe *Directory*, poiché non espone un metodo *GetAttribute*. Per ovviare al problema, si può utilizzare il metodo *File.GetAttributes* che può essere applicato anche alle directory. Un'altra possibilità consiste nell'utilizzare la proprietà *Attributes* di un oggetto *DirectoryInfo*.

PERCORSI ASSOLUTI O RELATIVI

Nei metodi e campi che accettano un percorso, è possibile fare riferimento ad un file o solo ad una directory. Il percorso può anche fare riferimento ad un percorso relativo, ad un percorso assoluto, ad un server e ad un nome di condivisione. Tutti i seguenti percorsi, ad esempio, sono accettabili:

"C:\MiaDir\MioFile.txt"
"C:\MiaDir"
"\\MioServer\MiaCondivisione"

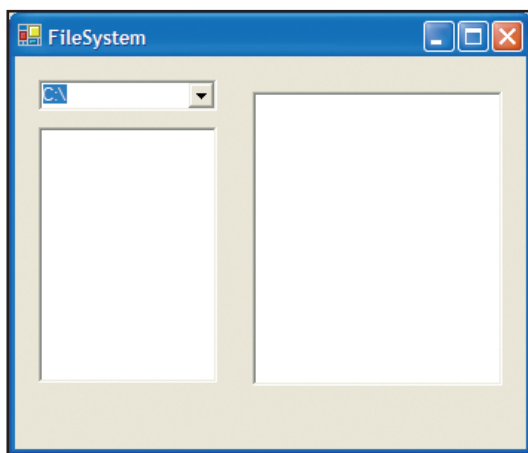
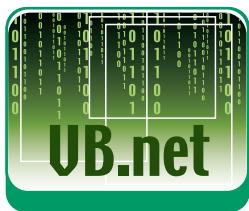



Fig. 2: ***In figura gli effetti del codice***

Nel momento in cui l'utente seleziona una unità logica, dobbiamo mostrare, in ListBoxDir, le directory contenute nel drive selezionato. Per questo motivo l'evento che dovrà contenere il codice necessario sarà l'evento *SelectedIndexChanged* del *ComboBox*

```
Private Sub
 ComboDrive_SelectedIndexChanged(ByVal sender As
          System.Object, ByVal e As System.EventArgs)
             Handles ComboDrive.SelectedIndexChanged

End Sub
```

Al solito definiamo una matrice di stringhe ed una variabile di tipo *String*

```
        Dim MatStrDir() As String
        Dim StrDir As String
```

In questo caso, si rende necessario introdurre un gestore di errori *Try..Catch* poiché si deve gestire l'errore di drive non pronto, che si verifica, ad esempio, quando si seleziona l'unità A:\ e nessun dischetto è stato inserito nell'unità.

```
       Try
             ........................
        Catch
MessageBox.Show("Driver non pronto")
        End Try
```

All'interno del gestore di errori, utilizziamo il metodo *GetDirectories*, passando come argomento il drive selezionato nel *ComboBox*, in modo da popolare la matrice di stringhe delle directory contenute nel drive selezionato

```
    MatStrDir = Directory.GetDirectories(
                                  ComboDrive.SelectedItem)
```

Utilizziamo il metodo *Clear* per pulire la *ListBox*

```
ListBoxDir.Items.Clear()
```

Infine con un ciclo *For Each .. Next* cicliamo sulla matrice delle directory ed aggiungiamo ogni elemento nella ListBox.

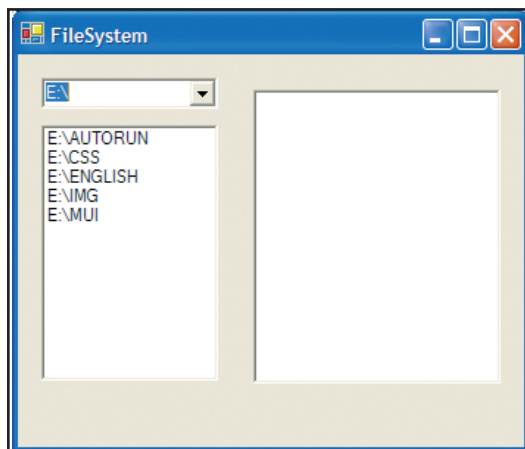



Fig. 3: ***In figura gli effetti del codice***

```
            For Each StrDir In MatStrDir
                ListBoxDir.Items.Add(StrDir)
            Next
```

L'ultima parte di codice sarà quella deputata a mostrare i file contenuti nella directory, nel momento in cui viene selezionata. Pertanto il codice necessario dovrà essere scritto nell'evento SelectedIndexChanged della prima ListBox.

```
Private Sub ListBoxDir_SelectedIndexChanged(ByVal
                      sender As System.Object, ByVal e As
                             System.EventArgs) Handles
                        ListBoxDir.SelectedIndexChanged

End Sub
```

Definiamo le solite variabili ed utilizziamo il metodo *GetFiles*, passando come argomento la directory selezionata nella *ListBox*, per popolare la matrice dei file.

```
        Dim MatStrFile() As String
        Dim StrFile As String
        MatStrFile =
              Directory.GetFiles(ListBoxDir.SelectedItem)
```

Infine utilizziamo il metodo *Clear* per pulire la *ListBox*, e con un ciclo *For Each .. Next* cicliamo sulla matrice dei file ed aggiungiamo ogni elemento in *ListBoxFile*.

```
        ListBoxFile.Items.Clear()
        For Each StrFile In MatStrFile
            ListBoxFile.Items.Add(StrFile)
        Next
```

LA CLASSE FILE

La gestione dei file, è garantita dalla classe *File*. La classe *File* permette la cancellazione, la copia o lo spostamento dei file e fornisce, inoltre, alcuni metodi che consentono di accedere alle informazioni di un file, come: la data di creazione, la data di modifica, gli attributi. Sono ancora disponibili, metodi che consentono di aprire un file e creare un oggetto *FileStream* utilizzato per le operazioni di lettura e scrittura su file.
Anche i metodi della classe *File* sono metodi statici:

- I metodi **SetAttributes** e **GetAttributes** permettono di impostare o di prelevare, un valore codificato in bit, che rappresenta una combinazione dei valori possibili degli attributi di un file (*Normal, Archive, ReadOnly, Hidden*, ecc)
- I metodi **SetCreationTime, SetLastAccessTime** e **SetLastWriteTime** permettono di modificare gli attributi relativi a data ed ora inerenti, rispettivamente, alla creazione, ultimo accesso e ultima scrittura di un file o una directory
- I metodi **GetCreationTime, GetLastAccessTime, GetLastWriteTime** restituiscono la data e l'ora inerenti, rispettivamente, alla creazione, ultimo accesso e ultima scrittura di un file.

Tornando alla nostra applicazione di esempio, quando l'utente seleziona un file in *ListBoxFile* possiamo mostrare a video le proprietà del file selezionato. L'evento che dovrà contenere il codice necessario sarà l'evento *SelectedIndexChanged* della *ListBox*.

```
Private Sub ListBoxFile_SelectedIndexChanged(ByVal
        sender As System.Object, ByVal e As
            System.EventArgs) Handles
        ListBoxFile.SelectedIndexChanged

End Sub
```

Molto semplicemente, dovremo utilizzare i metodi di *GetCreationTime*, *GetLastAccessTime*, *GetLastWriteTime*, passando come argomento il file selezionato nella *ListBox*, e comporre il messaggio da far comparire a video con il classico *MessageBox.Show*

```
    Dim DataCreazione As Date
    Dim DataModifica As Date
    Dim DataAccesso As Date
    DataCreazione =
     File.GetCreationTime(ListBoxFile.SelectedItem)
    DataModifica =
     File.GetLastWriteTime(ListBoxFile.SelectedItem)
```

```
    DataAccesso =
  File.GetLastAccessTime(ListBoxFile.SelectedItem)
    Dim Messaggio As String
    Messaggio = "File " & ListBoxFile.SelectedItem
                                          & vbCrLf
    Messaggio = Messaggio & "Data di Creazione :"
                         & DataCreazione & vbCrLf
    Messaggio = Messaggio & "Data Ultima
              Modifica: " & DataModifica & vbCrLf
    Messaggio = Messaggio & "Data Ultimo
               Accesso: " & DataAccesso & vbCrLf
    MessageBox.Show(Messaggio)
```

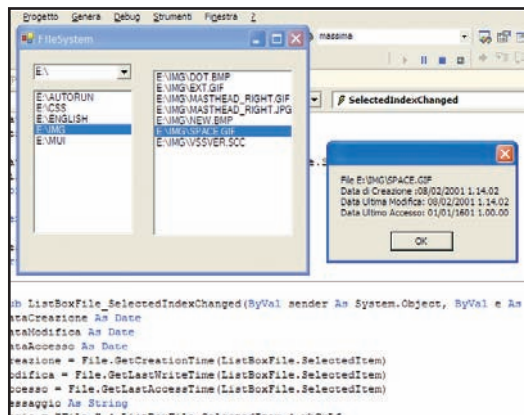
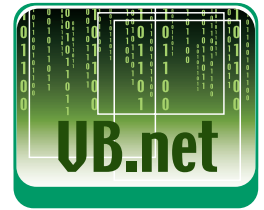



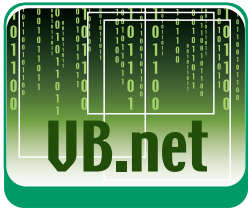
Fig. 5: *Il messaggio visualizzato dall'applicazione*

LETTURA E SCRITTURA DI UN FILE

Per le operazioni di lettura e scrittura, su un file aperto, si deve utilizzare il metodo *Open*. Il metodo *Open* apre un oggetto *FileStream* (oggetto di uno dei prossimi articoli), nel percorso specificato e con le modalità di accesso passate come argomento. Si devono impostare tre valori:

- **FileMode** - permette di indicare le modalità di

SecurityException	non si dispone dell'autorizzazione necessaria.
ArgumentException	il percorso specificato è una stringa di lunghezza zero, contiene solo spazi vuoti oppure uno o più caratteri non validi definiti da *InvalidPathChars*
ArgumentNullException	il percorso è Nothing
PathTooLongException	il percorso o le informazioni sul percorso assoluto superano la lunghezza massima definita dal sistema.
DirectoryNotFoundException	la directory specificata non esiste.
UnauthorizedAccessException	il percorso specificato fa riferimento ad un file che è in sola lettura e access non è Read. Si verifica anche nel caso in cui il percorso è una directory.
ArgumentOutOfRangeException	*mode* o *access* o *share* ha specificato un valore non valido.
FileNotFoundException	il file specificato non esiste.
NotSupportedException	il percorso contiene il carattere due punti (:) all'interno della stringa.

Tabella 1: Eccezioni di file.load

apertura di un file. Può assumere i valori: *Append, Create, CreateNew, Open, OpenOrCreate* e *Truncate*. Le modalità *Open* e *Append* falliscono nel caso il file non esista; *Create* o *CreateNew* falliscono se il file esiste già. Per aprire un file esistente si deve utilizzare *Open*. Per aprire un file o crearne uno se ancora non esiste si deve utilizzare *OpenOrCreate*. Per aggiungere un elemento ad un file, si deve utilizzare *Append*.

- **FileAccess -** permette di indicare le modalità di accesso in lettura e scrittura di un file. Può assumere i valori; *Read, Write, ReadWrite*.

- **FileShare -** permette di indicare il tipo di accesso al file di cui dispongono altri thread aperti. Può assumere i valori None (sono vietate tutte le operazioni), ReadWrite (sono consentite tutte le operazioni), Read, Write e Inheritable.

Esistono inoltre tre varianti del metodo Open che restituiscono un oggetto *FileStream*:

- **Create. -** Crea un file nel percorso completo specificato come argomento.
- **OpenRead. -** Apre un file esistente per la lettura.
- **OpenWrite.** - Apre un file esistente per la scrittura.

Infine, sono disponibili tre metodi che riguardano specificamente i file di testo: *CreateText, OpenText* e *AppendText* e si possono utilizzare per creare o aprire un file di testo o per aggiungere del testo in un file. Questi tre metodi restituiscono un oggetto *StreamReader* o un oggetto *StreamWriter*.

LA CLASSE DIRECTORYINFO

La classe *DirectoryInfo* rappresenta una singola directory e deriva dalla classe virtuale *FileSystemInfo*, come la classe *FileInfo*, che descriveremo in seguito. Proprietà e metodi interessanti sono:

- **Name -** restituisce il nome della directory.
- **FullName -** restituisce il nome completo della directory, incluso il relativo percorso.
- **Attributes -** permette di impostare od ottenere gli attributi della directory.
- **CreationTime, LastWriteTime** e **LastAccessTime** - permettono di impostare od ottenere la data e l'ora: della creazione, dell'ultima modifica o dell'ultimo accesso della directory

A differenza delle classi descritte finora, non è possibile utilizzare la classe *DirectoryInfo* ed i suoi metodi, senza aver ottenuto un riferimento ad un oggetto di tipo *DirectoryInfo*, per questo si deve adoperare il relativo metodo costruttore:

```
Public Sub New( ByVal Percorso As String)
```

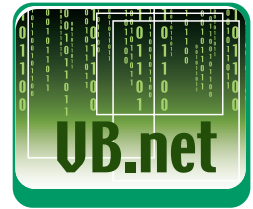
Dove l'argomento *percorso* accetta il percorso di una particolare directory.
La classe *DirectoryInfo* fornisce molti metodi analoghi a quelli della classe Directory, con la differenza che restituiscono degli oggetti *FileSystemInfo*, piuttosto che semplici stringhe.
Proviamo a riscrivere il codice dell'evento *SelectedIndexChanged* di *ComboDrive* utilizzando la classe *DirectoryInfo*, il risultato sarà:

```
Private Sub
 ComboDrive_SelectedIndexChanged(ByVal sender As
        System.Object, ByVal e As System.EventArgs)
          Handles ComboDrive.SelectedIndexChanged
      Dim MatFsiDir() As FileSystemInfo
      Dim FsiDir As FileSystemInfo
      Try
         ListBoxDir.Items.Clear()
         MatFsiDir = New
DirectoryInfo(ComboDrive.SelectedItem).GetDirectories
         For Each FsiDir In MatFsiDir
            ListBoxDir.Items.Add(FsiDir.FullName)
         Next
      Catch
         MessageBox.Show("Driver non pronto")
      End Try

End Sub
```

Possiamo subito notare le differenze:

- Le due variabili iniziali non sono più di tipo *String*, ma di tipo *FileSystemInfo*.
- Per valorizzare la matrice di oggetti *FileSystemInfo* abbiamo utilizzato il costruttore della classe *DirectoryInfo*, passando come argomento il drive selezionato in *ComboDrive* ed utilizzando l'analogo metodo *GetDirectories*.
- Per riempire *ListBoxDir*, abbiamo utilizzato il metodo *FullName* in modo da visualizzare il percorso completo delle sottodirectory.

LA CLASSE FILEINFO

La classe *FileInfo* rappresenta un singolo file. Analogamente a quanto detto per la classe *DirectoryInfo*, non è possibile utilizzare la classe *FileInfo* ed i suoi metodi, senza aver ottenuto un riferimento ad un oggetto di tipo *FileInfo*, per questo si deve adoperare il relativo metodo costruttore passando come argomento il nome completo del file:

```
Public Sub New( ByVal NomeFile As String)
```

Proprietà e metodi interessanti sono:

- **Name -** restituisce il nome del file.
- **FullName -** restituisce il nome completo del file, incluso il relativo percorso.
- **Attributes -** permette di impostare od ottenere gli attributi del file.
- **CreationTime**, **LastWriteTime** e **LastAccessTime -** permettono di impostare od ottenere la data e l'ora: della creazione, dell'ultima modifica o dell'ultimo accesso del file

Proviamo, infine, a riscrivere il codice, dell'evento *SelectedIndexChanged* di *ListBoxFile* utilizzando la classe *FileInfo*, il risultato sarà:

```
Private Sub ListBoxFile_SelectedIndexChanged(ByVal
                    sender As System.Object, ByVal e As
                          System.EventArgs) Handles
                      ListBoxFile.SelectedIndexChanged
    Dim DataCreazione As Date
    Dim DataModifica As Date
    Dim DataAccesso As Date
    Dim FsiFile As FileSystemInfo
    FsiFile = New FileInfo(ListBoxFile.SelectedItem)
    DataCreazione = FsiFile.CreationTime
    DataModifica = FsiFile.LastWriteTime
    DataAccesso = FsiFile.LastAccessTime
    Dim Messaggio As String
    Messaggio = "File " & FsiFile.Name & vbCrLf
    Messaggio = Messaggio & "Data di Creazione :"
                           & DataCreazione & vbCrLf
    Messaggio = Messaggio & "Data Ultima
                   Modifica: " & DataModifica & vbCrLf
    Messaggio = Messaggio & "Data Ultimo
                    Accesso: " & DataAccesso & vbCrLf
    MessageBox.Show(Messaggio)

End Sub
```

In questo caso le differenze sono:

- Utilizzo di una variabile di tipo *FileSystemInfo*.
- Per valorizzare la variabile, è stato utilizzato il costruttore della classe *FileInfo* passando come argomento il nome del file selezionato in *ListBoxFile*.
- Per comporre il messaggio riassuntivo, abbiamo utilizzato le proprietà *CreationTime*, *LastWriteTime* e *LastAccessTime*.

LA CLASSE PATH

La classe *Path* espone campi e metodi che consentono di ottenere informazioni su percorsi di file e directory. Il formato esatto di un percorso (così come l'estensione di un nome di file ed il set di caratteri permessi e vietati per separare gli elementi di un percorso), dipende dal sistema operativo utilizzato. La classe *Path* espone cinque campi statici che permettono di ottenere informazioni sui drive validi e sui separatori leciti per i nomi dei file, nel sistema operativo corrente.

- **AltDirectorySeparatorChar** restituisce un carattere alternativo utilizzato per separare i livelli di directory.
  - * *Path.AltDirectorySeparatorChar* restituisce /
- **DirectorySeparatorChar** restituisce il carattere utilizzato per separare i livelli di directory
  - * *Path.DirectorySeparatorChar* restituisce \
- **InvalidPathChars** restituisce una matrice di caratteri che non possono essere utilizzati nella definizione di un percorso.
  - * *Path.InvalidPathChars* restituisce "<>|
- **PathSeparator** restituisce il carattere separatore, usato per separare le stringhe di percorso nelle variabili di ambiente.
  - * *Path.PathSeparator* restituisce ;
- **VolumeSeparatorChar** restituisce il carattere separatore di volume.
  - * *Path.VolumeSeparatorChar* restituisce :

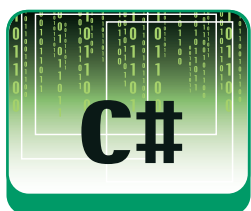
La classe *Path* fornisce, inoltre, dei metodi che permettono di estrarre informazioni dal percorso di un file senza preoccuparsi che il file o la directory esistano o meno. Se, ad esempio, applichiamo tali metodi al file C:\MiaDir\MioFile.txt otteniamo:

- **GetDirectoryName -** restituisce le informazioni sulla directory quindi *C:\MiaDir*
- **GetFileName -** restituisce il nome e l'estensione del file quindi *MioFile.txt.*
- **GetExtension -** restituisce l'estensione del file quindi *.txt.*
- **GetFileNameWithoutExtension -** restituisce il nome del file senza estensione quindi *MioFile*
- **GetPathRoot** - restituisce la directory principale quindi C:\.
- **HasExtension -** restituisce *True* se un percorso include un'estensione di nome di file.
- **IsPathRooted** - restituisce *True* se il percorso specificato contiene informazioni su un percorso assoluto altrimenti, in caso di percorso relativo, restituisce *False*

Segnaliamo infine i metodi *GetTempPath* e *GetTempFileName* che non accettano argomenti e restituiscono rispettivamente la posizione della directory temporanea in Windows ed il nome di un file temporaneo.

Luigi Buono

Le mani nel registro con .NET

La libreria di base di .NET mette a disposizione gli strumenti necessari per accedere al registro di sistema. Ma in alcuni casi è necessario ricorrere alle API di Windows

Il registro di Windows e il repository fondamentale nel quale vengono custodite tutte le impostazioni di sistema, sia che riguardino le applicazioni installate, i parametri di funzionamento del sistema operativo o le periferiche installate e così via. Questo database, nonostante la sua fondamentale importanza, è strutturato in maniera molto semplice. Ogni parametro è una chiave a cui può essere associato un valore. Tutte le chiavi sono organizzate in un albero, le cui radici sono le chiavi principali, e da cui si diramano tutte le altre, a seconda della loro categoria. I rami principali del registro, che discendono dalle radici dell'albero, sono anche detti hive. Tali hive sono dunque un gruppo di chiavi, sottochiavi e valori che dispone di una serie di file di supporto contenenti i backup dei dati relativi.

L'ALBERO DELLE CHIAVI

Le chiavi fondamentali del registro sono sei, come riepilogato nel seguente elenco e mostrato nel digramma.
Come potete notare dalla **Figura 1**, il loro nome inizia con la sigla *HKEY*, che sta per *Handle Key*. Vediamo il significato e dunque qual è il contenuto di ogni sottoramo che discende da queste chiavi radice.

Fig. 1: La struttura del registro di Windows

- **HKEY_CLASSES_ROOT:** Il ramo *HKCR* contiene i nomi di tutti i tipi di file registrati nel sistema, e le loro proprietà. È grazie a queste informazioni, ad esempio che, al doppio click su uno specifico file, viene aperta l'applicazione che permette di visualizzare e trattare correttamente il file.
- **HKEY_CURRENT_USER:** Questo hive contiene le informazioni di configurazione dell'utente che è loggato nel sistema, ad esempio le impostazioni del desktop, il percorso della cartella documenti, le applicazioni che può eseguire.
- **HKEY_LOCAL_MACHINE:** Contiene le configurazioni della macchina, comune a tutti gli utenti.
- **HKEY_USERS:** Questo ramo contiene tutti i profili degli utenti del sistema.
- **HKEY_CURRENT_CONFIG.** Il ramo *HKCR* contiene le informazioni sull'hardware installato sul sistema.
- **HKEY_DYN_DATA.** Questa chiave esiste solo sui sistemi Windows 9x/ME, e contiene dati dinamici, e che variano frequentemente.

In realtà esiste un'altra chiave di primo livello, la *HkeyPerformanceData.*

- **HKEY_PERFORMANCE_DATA.** Contiene informazioni sulle performance dei componenti software in esecuzione nel sistema.

Sebbene ciò possa sembrare complicato, la struttura è totalmente equivalente all'albero delle directory del file system, che si dirama da una unità disco, ad esempio da C:, D: e così via, per ogni unità logica.

VALORI DELLE CHIAVI

Così come una directory contiene anche file, ogni chiave del registro può contenere valori, di 5 tipi fondamentali.

- **REG_BINARY** – memorizza un valore in formato binario che viene visualizzato in genere in formato esadecimale.
- **REG_DWORD** – rappresenta un numero intero formato da quattro byte.
- **REG_EXPAND_SZ** – è una stringa contenente una variabile d'ambiente, espandibile da una applicazione, ad esempio *"%SystemRoot%"*.
- **REG_MULTI_SZ** – rappresenta una lista di stringhe.
- **REG_SZ** – questo tipo è una stringa standard, usata quindi per rappresentare testi leggibili.

Altri tipi di valori sono possibili, ma solitamente gli editor di registro, come il classico regedit, non permettono di crearli e visualizzarli. Al termine di questo articolo, dovreste però essere in grado anche di trattare un valore di tipo *QWORD*, cioè un numero a 64bit, valori indefiniti *REG_NONE*, ed altri ancora.

UN CLONE DI REGEDIT

Il .NET Framework mette a disposizione le classi necessarie ad interagire con il registro di sistema, per la creazione, la modifica, la lettura di chiavi e valori. Per esplorare ogni possibilità realizzeremo allora un nostro editor del registro. La **Figura 2** mostra l'applicazione completa e funzionante, e da cui si può notare la totale somiglianza con l'editor regedit fornito da Windows.
In allegato alla rivista trovate tutto il codice, oltre all'applicazione funzionante. Fate pure riferimento ad esso per quanto riguarda la realizzazione dell'interfaccia grafica, dato che essa non è fra gli scopi dell'articolo e non ne parleremo. Ma per chi è alle prime armi con

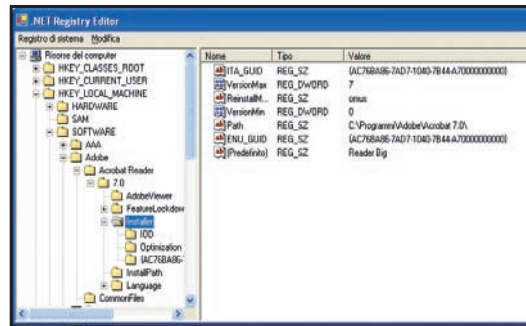



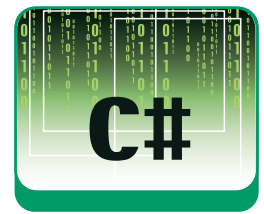
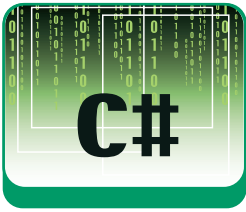
Fig. 2: ***Un clone del regedit in c#***

TreeView e *ListView* può essere comodo darci un'occhiata.

UNA CLASSE PER IL REGISTRO

Sebbene il .NET Framework metta a disposizione quasi tutto il necessario per lavorare con il registro, in alcuni casi è necessario ricorrere alle API di Windows. In ogni caso, parlando di programmazione orientata agli oggetti, è bene strutturare un minimo le classi che faranno parte dell'applicazione. Per permettere un accesso al registro più semplice da capire e facile da utilizzare dalle classi della GUI, è stata dunque creata una classe *WindowsRegistry*, dalla quale vengono usate le classi del namespace *Microsoft.Win32*, ed una classe per l'accesso alle funzioni native del registro, che si chiama *RegAPI*, utilizzata anch'essa dalla classe *WindowsRegistry.* Inoltre la classe *WindowsRegistry* fornirà metodi di utilità, ad esempio per ottenere un nome non esistente alla creazione di una nuova chiave, o di un nuovo valore. Un esempio di accesso al registro tramite le API, è il caso di creazione o modifica di valori con tipi diversi da *REG_SZ*, *REG_DWORD*, o *REG_BINARY.* In un caso come la creazione di un valore *REG_MULTI_SZ* è necessario utilizzare la funzione *RegSetValueEx* della *advapi32.dll*, importandola in C# così:

```
[DllImport("advapi32.dll", CharSet =
    CharSet.Unicode, EntryPoint = "RegSetValueEx")]
public static extern int RegSetValueEx(
            IntPtr hKey,
            string lpValueName,
            int Reserved,
            int    dwType,
            string lpData,
            int cbData
);
```

Vedremo più avanti come utilizzare il metodo per creare o modificare un valore.

RICREARE IL REGISTRO

Per ricreare il registro basta partire dalle chiavi di primo livello che abbiamo già visto, e ricavare, ogni volta che l'utente richiede di espandere un ramo e le sottochiavi del livello sottostante, scendendo mano a mano di livello. Naturalmente per visualizzare l'albero bisogna utilizzare un controllo *TreeView*.
Il livello delle radici verrà creato dal metodo seguente:

```
private void BuildRegistryTree()
{
    treeView.Nodes.Clear();
    treeView.ImageList=treeImageList;
    TreeNode root=new TreeNode("Risorse del
                                computer",0,0);
    TreeNode node=new TreeNode(
                Registry.ClassesRoot.Name,1,2);
    node.Tag=Registry.ClassesRoot;
    root.Nodes.Add(node);
    node=new TreeNode(
                Registry.CurrentUser.Name,1,2);
    node.Tag=Registry.CurrentUser;
    root.Nodes.Add(node);
    node=new TreeNode(
                Registry.LocalMachine.Name,1,2);
    node.Tag=Registry.LocalMachine;
    root.Nodes.Add(node);
    node=new TreeNode(
                Registry.Users.Name,1,2);
    node.Tag=Registry.ClassesRoot;
    root.Nodes.Add(node);
    node=new TreeNode(
                Registry.CurrentConfig.Name,1,2);
    node.Tag=Registry.CurrentConfig;
    root.Nodes.Add(node);
    treeView.Nodes.Add(root);

}
```

Ogni nodo conterrà nella proprietà *Tag* l'oggetto *RegistryKey*, in maniera da poterlo riutilizzare facilmente quando dovremo espandere un ramo. Quando l'utente seleziona una chiave, sia essa radice che una di più basso livello, verrà scatenato l'evento *AfterSelect* del controllo *TreeView*.

PINVOKE

Per utilizzare una funzione definita in una DLL nativa del sistema operativo, bisogna utilizzare l'attributo DLLImport, indicando il nome della dll e convertendo nei giusti tipi di .NET, parametri. Ad esempio, per importare la funzione *RegOpenKeyEx*, definita in *advapi32.dll* scriveremo:

```
[DllImport("advapi32.dll", CharSet =
        CharSet .Unicode, EntryPoint =
                    "RegOpenKeyEx")]
public static extern int RegOpenKeyEx(
IntPtr hKey, string subKey, uint options,
        int sam, out IntPtr phkResult );
```

All'interno del gestore, si espanderà il nodo se è la prima volta che l'utente seleziona la chiave selezionata, e si mostreranno nella *ListView* di destra, i valori eventuali che la chiave contiene.

```
private void treeView_AfterSelect(object sender,
        System.Windows.Forms.TreeViewEventArgs e)
{
  this.Cursor=Cursors.WaitCursor;
  if(e.Node != null && e.Node.Tag!=null &&
                    e.Node.Tag is RegistryKey )
  {
    if(e.Node.Nodes.Count==0)
    {
      ExpandKeyNode(e.Node);
    }
    RegistryKey key=(RegistryKey)e.Node.Tag;
    ShowKeyValues(key);

  }
  this.Cursor=Cursors.Default;
}
```

Il metodo *ExpandKeyNode* è quello che si occupa di ricavare le sottochiavi.

```
public void ExpandKeyNode(TreeNode parent)
{
  TreeNode node=new TreeNode();
  RegistryKey key=parent.Tag as RegistryKey;
  string[] subKeys=key.GetSubKeyNames();
  for(int i=0,len=key.SubKeyCount;i<len;i++)
  {
    try
    {
      node=new TreeNode(subKeys[i],1,2);
      node.Tag=key.OpenSubKey(subKeys[i]);
      parent.Nodes.Add(node);
    }
    catch(System.Security.SecurityException sExc)
    {
      //ignore
    }
  }
}
```

Per realizzare ciò viene invocato il metodo *GetSubKeyNames* della classe *RegistryKey*, e per ogni nome di chiave così ottenuto, la sottochiave viene aperta, con il metodo *OpenSubKey* ed inserita come nuovo nodo dell'albero e nella relativa proprietà *Tag*.
Per mostrare invece i valori delle chiavi selezionate, viene invocato il metodo *ShowKeyValues*, all'interno del quale tramite il metodo *GetValueNames* della classe *RegistryKey* vengono ricavati i nomi dei valori, e per ognuno

viene aggiunto un item alla *ListView* mediante il metodo *CreateRegistryValue:*

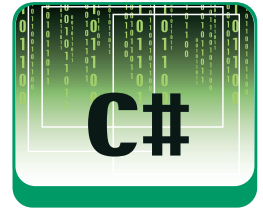
```
private ListViewItem CreateRegistryValue(
                                RegistryKey key,string val)
{
   ListViewItem itm=new ListViewItem();
   if(val==String.Empty)
      itm.Text="(Predefinito)";
   else itm.Text=val;
   WindowsRegistry wreg=new WindowsRegistry();
   RegistryType valType=wreg.GetValueType(key,val);
   object obj=key.GetValue(val);
   string type="";
   if(valType==RegistryType.REG_EXPAND_SZ)
   {
      itm.SubItems.Add(
            RegistryType.REG_EXPAND_SZ.ToString());
      itm.ImageIndex=0;
      itm.Tag=obj.ToString();
      itm.SubItems.Add(obj.ToString());
   }
   if(obj is string)
   {
      itm.SubItems.Add(
                  RegistryType.REG_SZ.ToString());
      itm.ImageIndex=0;
      itm.Tag=obj.ToString();
      itm.SubItems.Add(obj.ToString());
   }
   else if(obj.GetType().FullName=="System.Int32")
   {
     itm.SubItems.Add(
             RegistryType.REG_DWORD.ToString());
     itm.SubItems.Add(obj.ToString());
     itm.ImageIndex=1;
   }
   else if(obj.GetType().FullName=="System.Byte[]")
   {
     byte[] valori=(byte[])obj;
     itm.SubItems.Add(
             RegistryType.REG_BINARY.ToString());
     string valBin="(valore binario vuoto)";
     for(int i=0,len=valori.Length;i<len;i++)
     {
       if(i==0)
         valBin=String.Format("{0:x2}",valori[i]);
       else
         valBin+=String.Format(" {0:x2}",valori[i]);
     }
     itm.SubItems.Add(valBin);
     itm.ImageIndex=1;
   }
   else if(obj.GetType().FullName=="System.String[]")
   {
      string[] valori=(string[])obj;
      itm.SubItems.Add(
             RegistryType.REG_MULTI_SZ.ToString());
      string valStr="";
      for(int i=0,len=valori.Length;i<len;i++)
      {
        valStr+=String.Format("{0}",valori[i]);
        if(i<len-1)
          valStr+="\r\n";
      }
      itm.SubItems.Add(valStr.Replace("\r\n"," "));
      itm.Tag=valStr;
      itm.ImageIndex=0;
   }
   return itm;
}
```

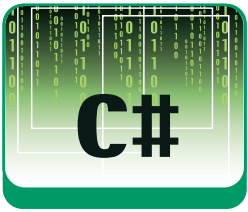
Il metodo è uno dei più lunghi, in quanto a seconda del tipo del valore letto, deve distinguere il suo comportamento. Il metodo *GetValue* della classe *RegistryKEy*, restituisce un object generico, del quale è necessario quindi ottenere il tipo.

I tipi restituiti possibili sono tipo string, *byte[]*, *int*, oppure *string[]*. Rispettivamente essi dunque ci diranno se il valore è un *REG_SZ*, un valore *REG_BINARY*, un *REG_ DWORD*, o un *REG_MULTISZ*. Però anche il tipo *REG_EXPAND_SZ* verrà riconosciuto come una stringa qualunque, e quindi è necessario in questo caso utilizzare le API di Windows per riconoscere correttamente il tipo.

La classe *WindowRegistry* implementa dunque allo scopo il metodo *GetValueType:*

```
public RegistryType GetValueType(
                        RegistryKey key,string valName)
{
     IntPtr hkey;
     RegistryKey khive=GetRegistryHive(key.Name);
     string kname=key.Name.Substring(
                           key.Name.IndexOf("\\")+1);
     uint hive=RegAPI.RegistryHKeyToNative(khive);
     int lResult = RegAPI.RegOpenKeyEx(
         new IntPtr((int)hive),
         kname,
         0, (int)RegAPI.KEY_QUERY_VALUE, out hkey);
    if(lResult==0)
    {
         int lpType=0,data=0;
         System.Text.StringBuilder sb=new
                           System.Text.StringBuilder();
         lResult=RegAPI.RegQueryValueEx(
              hkey,valName,null,ref lpType,null,ref data);
       return (RegistryType)lpType;
    }
    return RegistryType.UNKNOWN;
}
```

Esso ricorre alle funzioni native *RegOpenKeyEx* e *RegQueryValueEx* per ottenere il tipo.

REGEDIT

L'editor del registro di Windows non ha un collegamento nel menu dei programmi installati. Per avviarlo è necessario dunque lanciarlo manualmente, cliccando su *Start* poi su *Esegui…* e digitando il nome *regedit*.

I tipi vengono rappresentati come membri dell'enumerazione *RegistryType:*

```
public enum RegistryType
  {
    UNKNOWN=-1,
    REG_SZ=1,
    REG_BINARY=3,
    REG_DWORD=4,
    REG_MULTI_SZ=7,
    REG_EXPAND_SZ=2
  }
```

Non sono tutti i valori possibili, ma per il nostro editor, così come per il classico regedit, sono quelli effettivamente utilizzati.

BIBLIOGRAFIA

- **C# AND .NET PLATFORM** ***Troelsen*** **(APress)**
- **APPLIED MICROSOFT .NET PROGRAMMING** ***Richter*** **(Microsoft Press)**
- **ESSENTIAL .NET, VOLUME I: THE COMMON LANGUAGE RUNTIME** ***Don Box , Chris Sells*** **(Addison Wesley)**

CREARE UNA CHIAVE

Per creare una chiave di registro, bisogna avere a disposizione una chiave di registro esistente, sulla quale invocare il metodo *OpenSubKey* con il parametro *writable* pari a *true*. Data dunque una qualunque chiave del registro, rappresentata da un nodo dell'albero, è priorità indispensabile ottenere la radice dell'albero, cioè il *RegistryHive,* sotto forma di *RegistryKey.* La classe *TreeNode* permette di ricavare il path completo del nodo, e da questo possiamo dunque ottenere l'*hive* relativo.
Basta implementare un metodo che verifichi tutte le radici possibili fino a trovare quella giusta.
Il metodo *GetRegistryHive* seguente ottiene questo obiettivo:

```
private RegistryKey GetRegistryHive(string path)
{
  if(path.IndexOf(Registry.ClassesRoot.Name)>=0)
  {
    return Registry.ClassesRoot;
  }
  else if(path.IndexOf(
                    Registry.CurrentConfig.Name)>=0)
  {
    return Registry.CurrentConfig;
  }
  else if(path.IndexOf(
                    Registry.CurrentConfig.Name)>=0)
  {
    return Registry.CurrentUser;
  }
  else if(path.IndexOf(
                    Registry.LocalMachine.Name)>=0)
  {
    return Registry.LocalMachine;
  }
  else if(path.IndexOf(Registry.Users.Name)>=0)
  {
    return Registry.Users;
  }
  return null;
}
```

a questo punto, avendo la chiave radice, basta aprire la chiave sotto la quale creare la sotto chiave ed invocare il metodo *CreateSubKey* di *RegistryKey:*

```
RegistryKey hive=GetRegistryHive(path);
RegistryKey key=hive.OpenSubKey(
            path.Substring(hive.Name.Length+1),true);
key.CreateSubKey(name);
```

CREARE I VALORI

L'ultima funzionalità mancante all'applicazione è la creazione dei valori di una chiave.
Nel menu *"modifica"* sono state aggiunte le diverse voci per creare un nuovo valore, uno per ogni tipo supportato, come è possibile vedere in **Figura 3**.

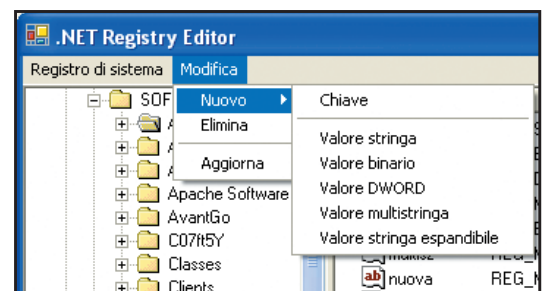
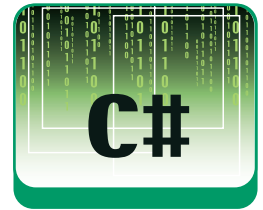



Fig. 3: ***Il menu per la creazione di un valore***

Per creare un valore, la classe *WindowsRegistry* fornisce il metodo *SetValue* che, a seconda del tipo del valore da creare, invoca il metodo *SetValue* di *RegistryKey,* oppure in caso di un tipo non supportato utilizza le API di Windows.
Il tipo di valore da creare dipende dal valore *object* utilizzato al momento di impostarne il valore.
Quindi se si vuole creare un valore *DWORD* bisogna effettuare un cast verso il tipo *int,* per un binary occorre un array *byte[],* per una stringa multilinea un array di stringhe e così via.

```
public void SetValue(RegistryKey key,string name,
                        object val, RegistryType type)
{
  try
  {
    RegistryKey hive=GetRegistryHive(key.Name);
```

```
    RegistryKey wkey=hive.OpenSubKey(
            key.Name.Substring(key.Name.IndexOf(
                                "\\")+1),true);
    switch(type)
    {
      case RegistryType.REG_MULTI_SZ:
      {
         string s=val.ToString().Replace("\r","");
         string[] msz=s.Split('\n');
         wkey.SetValue(name,msz);
         break;
      }
      case RegistryType.REG_EXPAND_SZ:
      {
         SetValueNative(wkey,name,
                val,RegistryType.REG_EXPAND_SZ);
         break;
      }
      case RegistryType.REG_SZ:
      {
         wkey.SetValue(name,val);
         break;
      }
      case RegistryType.REG_DWORD:
      {
         wkey.SetValue(name,(int)val);
         break;
      }
      case RegistryType.REG_BINARY:
      {
         wkey.SetValue(name,(byte[])val);
         break;
      }
      default:
      break;
    }
    wkey.Close();
  }
  catch
  {
    throw;
  }
}
```

Notate che dopo l'apertura di una chiave è importante invocare il metodo *Close* per chiuderla.
Come detto, nel caso di un valore di tipo *REG_EXPAND_SZ* viene invocato il metodo *SetValueNative*, che appunto usa le funzioni native di Windows:

```
public void SetValueNative(RegistryKey key,
         string valName,object val, RegistryType type)
{
  IntPtr hkey;
  RegistryKey khive=GetRegistryHive(key.Name);
  string kname=key.Name.Substring(
```

```
                        key.Name.IndexOf("\\")+1);
  uint hive=RegAPI.RegistryHKeyToNative(khive);
  int lResult = RegAPI.RegOpenKeyEx(
    new IntPtr((int)hive),
    kname,
    0, (int)RegAPI.KEY_SET_VALUE, out hkey);
  if(lResult==0)
  {
    switch(type)
    {
      case (RegistryType)RegAPI.REG_EXPAND_SZ:
      RegAPI.RegSetValueEx(hkey,
      valName,
      0,(int)type,
      val.ToString(),
      val.ToString().Length*2);
       break;
       default:break;
    }
  }

}
```

Le funzioni native sono esposte come detto dalla classe *RegAPI*, utilizzando la tecnologia *Platform Invoke* per importare le funzioni e le costanti necessarie.
Ad esempio i tipi dei valori hanno dei valori interi corrispondenti, che potete ricavare, se avete il platform SDK installato, dando un'occhiata agli header file *winnt.h* e *winreg.h.*

```
public const uint REG_NONE =0x00000000;
public const uint REG_SZ =0x00000001;
public const uint REG_EXPAND_SZ =0x00000002;
public const uint REG_BINARY =0x00000003;
public const uint REG_DWORD=0x00000004;
public const uint REG_MULTI_SZ=0x00000007;
```

CONCLUSIONI

In questo articolo abbiamo visto come realizzare un clone del RegEdit di Windows, allo scopo di utilizzare tutte le classi e i metodi messi a disposizione dal .NET Framework.
Inoltre abbiamo anche utilizzato la tecnologia PInvoke per invocare le API di Windows che riguardano le stesse funzionalità.
Vogliamo ricordare che il registro di Windows costituisce una componente delicata del sistema. Per cui, per fare qualche prova con esso è bene prestare la massima attenzione, magari creando un backup del registro prima di modificarlo o magari utilizzando chiavi e valori creati ad hoc, per evitare di rendere il sistema addirittura inutilizzabile.

Antonio Pelleriti

Potete contattare l'autore per suggerimenti, critiche o chiarimenti all'indirizzo e-mail antonio.pelleriti@ioprogrammo.it **oppure sul sito** www.dotnetarchitects.it

Configurare Windows via codice

Con diversi esempi scopriamo i segreti di WMI (Windows Management Instrumentation) lo strumento che permette di controllare le caratteriste principali del sistema

WMI - *Windows Management Instrumentation* (Strumentazione gestione Windows) è un potente strumento presente nei sistemi operativi *Windows*. *WMI* permette di ritrovare ed impostare le informazioni caratteristiche dell'*Hardware* e del *Software* di computer locali o remoto. *WMI* è installato in *Windows 2003/XP/ME/2000*, mentre per i restanti sistemi deve essere scaricato dal sito della *Microsoft*. *WMI* può essere programmato con diversi linguaggi tra i quali *Visual Basic*, *VBScript* e *WSH*. Lo sviluppatore può utilizzare *WMI* per svariati scopi: dalla ricerca d'informazioni sui componenti e le periferiche del sistema alla definizione di particolari *task* di gestione e controllo degli eventi. Per la programmazione di *WMI* con gli *Script* e con *Visual Basic* sono disponibili le *Scripting API for WMI*. In *Windows 2000* e versioni superiori alcune caratteristiche di *WMI* possono essere gestite, dagli utenti, con il *Controllo WMI*. Nell'articolo tratteremo i seguenti argomenti: le caratteristiche principali del *WMI* e della sua architettura; i *Provider* e le classi *WMI*; il modello ad oggetti per *WMI* e la libreria *Wbemdisp.tlb* per *Visual Basic*. Come esempi, presenteremo alcuni *Script* e introdurremo un'applicazione che permette di ricavare ed impostare diverse proprietà del sistema operativo. In particolare vedremo come impostare la stampante predefinita e come gestire i processi in esecuzione nel sistema; inoltre mostreremo come interrogare le classi *WMI* con delle query scritte in *WQL* un linguaggio simile all'*SQL*.

L'ARCHITETTURA DEL WMI

Il *WMI* è l'implementazione delle specifiche *WBEM* (*Web-Based Enterprise Management*) che puntano alla definizione di uno standard per l'accesso e la condivisione d'informazioni di gestione sulle reti aziendali. Il *Core* di *WBEM* è il *CIM* (*Common Information Model*) che è il modello di dati per *WBEM*, *CIM* fornisce un modello per la definizione dei dati per il *sistema*, il *network*, le applicazioni

Fig. 1: **Il Controllo WMI e gli Spazi dei nomi**

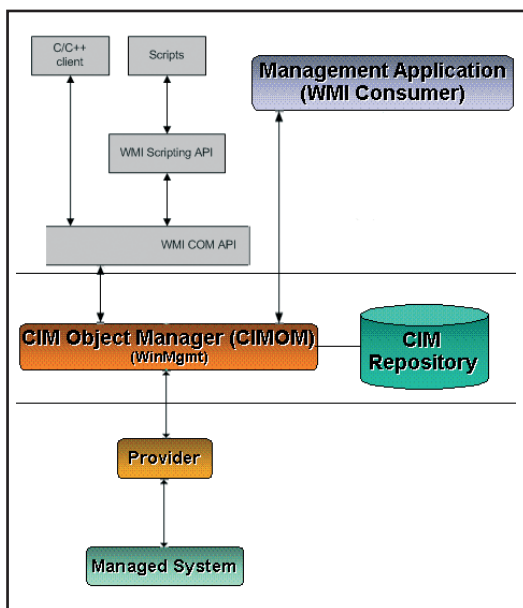



Fig. 2: **La struttura a tre livelli di WMI**

REQUISITI

Conoscenze richieste
Conoscenze di base sulla gestione dei file, su SQL e sulla ListView.

Software
Piattaforma Windows 98 o superiore Visual Basic 6 SP6.

Impegno

Tempo di realizzazione

ed i *servizi*. Queste informazioni sono archiviate nel *CIM Repository*.
In **Figura 2** riportiamo una descrizione, secondo il modello a tre livelli (*three-layer*), dell'architettura del *WMI*. Nel modello partendo dal più basso livello distinguiamo: *Provider*, *CIMON* e *Consumer*.

1. I **Provider** sono gli intermediari tra il sistema, che deve essere controllato, e gli oggetti *CIM*. In altre parole i *Provider* si preoccupano di mappare gli eventi ed i dati degli oggetti controllati (*Manager System*) nelle corrispondenti classi *CIM*.

2. **CIMON (CIM Object Manager)** tiene traccia delle classi disponibili e del *Provider* che fornisce le loro istanze, la definizione delle classi è contenuta nel *Repository*. Nel sistema operativo il *CIMON* ed il *Repository* sono gestiti con il servizio *WinMgmt*.

3. Il livello più alto è rappresentato dai **Consumer** cioè dalle applicazioni. Per esempio uno *Script*, un programma *Visual Basic* oppure un *tool* di alto livello del sistema come il *Controllo WMI*. Questi *Consumer* hanno bisogno soltanto di conoscere quale classe fornisce le informazioni, non è rilevante sapere da dove arrivano i dati, cioè conoscere il *Provider*.

Per quanto riguarda il Servizio *WinMgmt* (*Windows Management Service*) dobbiamo aggiungere che in *Windows 98* è costituito dall'eseguibile *Winmgmt.exe*, mentre in *Windows XP* è implementato nel processo *SVCHOST*.

PROVIDER E CLASSI

Sicuramente questo paragrafo, non può essere considerato esaustivo sul complesso mondo dei *Provider* e delle classi *WMI*, per questo prendetelo come un ausilio per future esplorazioni. Sicuramente uno dei più noti *Provider* di *WMI* è *Win32* (file *cimwin32 .dll*), relativo allo spazio dei nomi di *default* (*root \cimv2*). Esso fornisce informazioni sul sistema operativo, il computer, le periferiche, i *file systems* e la sicurezza. Nei nostri esempi abbiamo utilizzato diverse classi di questo *Provider*; in tabella 1 riportiamo la descrizione di alcune di esse. Facciamo notare che per identificare una classe relativa ad un particolare *Provider* bisogna utilizzare una stringa come la seguente *nomeprovider_nomeclasse*

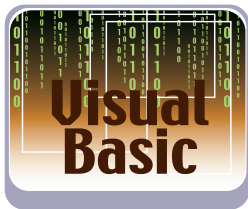
Nome classe	Descrizione
Printer	Rappresenta un dispositivo collegato ad un computer Win32 in grado di riprodurre dati su carta.
Process e ProcessStartup	Per gestire i processi, ricavare informazioni su di essi, modificarli e per crearne dei nuovi.
LogicalDisk	Utile per ricavare informazioni sui dischi nel sistema anche se rimovibili o mappati.
SystemBIOS	Contiene diverse informazioni sul Bios: avvio, fuso orario, lingua, descrizione.
VideoController	Rappresenta le caratteristiche e la capacità di gestione del controller video.
PhysicalMemory	Racchiude informazioni sulle periferiche di memoria fisica situate su un computer e disponibile per il sistema operativo.
OperatingSystem	Racchiude informazioni sul sistema operativo.
Processor	Fornisce informazioni sul tipo di processore, velocità ecc.

Tabella 1: Alcune classi di Win32

per esempio *Win32 ProcessStartup*. Descriviamo brevemente altri *Provider* presenti sul computer. *Registry Provider* (*stdprov.dll*), relativo al *NameSpace root\default*, permette di creare chiavi di registro, di leggerle e modificarle, inoltre permette di generare degli eventi *WMI* allorché sono modificati alcuni elementi del registro. *Windows Installer Provider* (*msiprov.dll*), relativo al *NameSpace root\ cimv2*, consente il controllo di *Windows Installer*. *Active Directory Provider* (*dsprov .dll*), relativo al *NameSpace root\directory \ldap*, permette di gestire le informazioni relative a *Microsoft Active Directory*. Se volete consultare l'elenco completo delle classi *WMI* vi consigliamo di scaricare *WMI Tools* dal sito della *Microsoft*.

OGGETTI WMI

Per accedere alle classi del *WMI* dobbiamo conoscere alcuni oggetti del *WMI Scripting Objects*, che è il modello ad oggetto che permette a *Visual Basic* e agli *Script* di interagire con *WMI*. In particolare, gli oggetti che utilizzeremo nei nostri esempi sono i seguenti: *SWbemServices*, *SWbemLocator*, *SWbemObject* e *SWbemObjectSet* - una collezione di oggetti *SWbemObject*. L'oggetto *SWbemServices* permette di creare, modificare e cercare istanze di classi. Questo oggetto presenta diversi metodi e la proprietà *Security*, da usare per impostare il livello di sicurezza. Dei metodi, invece, utilizzeremo: *ExecQuery(queryWQL)* che permette di eseguire una query *WQL*; *InstancesOf* (*nomeclasse*) che restituisce una collezione di oggetti della classe specificata come parametro e *Get(Path oggetto)* che restituisce l'istanza di un singolo oggetto. Facciamo notare, però, che l'oggetto *SWbemServices* non può essere creato con la *Create-*

WSH
***Windows Script Host* – WSH - permette di eseguire Script dentro i sistemi operativi *Windows a 32-bit*. WSH, essendo basato sulla tecnologia *Activex Scripting*, può essere visto come un contenitore di oggetti *COM*. La versione più recente di *WSH*, fornita con Windows XP, è la 5.6. Le altre versioni sono la 1.0, che si trova in Windows 98 e Windows NT e la 2.0 fornita con Windows 2000. Dal sito http://msdn.microsoft.com/scripting è possibile fare l'upgrade alla versione 5.6.**

Object. *SWbemLocator*, invece, è l'oggetto che ricava un *SWbemServices* connesso ad un *namespace* su un computer locale o remoto. L'oggetto ha il metodo *ConnectServer* che appunto connette al *WMI* di un computer. Infine *SWbemObject* permette di referenziare e manipolare l'istanza di una classe *WMI*.

INTERAGIRE CON WMI

In questo paragrafo descriviamo le operazioni di base per utilizzare il *servizio WMI* cioè per interagire con *WinMgmt*. Come accennato l'interazione può essere fatta con uno *Script* o direttamente con *Visual Basic* referenziando la libreria *Wbemdisp*. In entrambi i casi essa può avvenire secondo i seguenti punti.

1. Individuare il *servizio WMI* sul computer locale o remote.
2. Connettersi al servizio con un certo livello di sicurezza.
3. Selezionare uno spazio dei nomi (*NameSpace*), un *Provider* e una classe.
4. Eseguire una *query WQL* o impostare dei valori.

I primi tre punti, quando si lavora sul computer locale, con il *NameSpace* e con i livelli di sicurezza di *default* possono essere considerati come un'unica operazione. Le istruzioni per impostare i punti 1 e 2 con uno *Script*, sono le seguenti:

```
Set services = GetObject("winmgmts:")
services.Security_.ImpersonationLevel = 3
```

Dove *"winmgmts:"* è un *moniker* cioè un identificatore (o meglio uno *short name*) che serve per specificare un *namespace*, una classe o una sua istanza. Il *moniker winmgmts*, nel nostro caso usato senza parametri, dice, a *Windows Script Host*, che si vuole lavorare sul computer locale e che si vuole utilizzare lo spazio dei nomi di *default*. La seconda istruzione, invece, imposta il livello di sicurezza su quello di *default* (o meglio *impersonate*).
In *Visual Basic* - con la libreria *Wbemdisp* - per le stesse azioni utilizziamo le seguenti:

```
Dim objWMIService As SWbemServices
Dim objLocator As SWbemLocator
Set objLocator = New SWbemLocator
Set objWMIService = objLocator.ConnectServer()
objWMIService.Security_.ImpersonationLevel = 3
```

Anche in questo caso, *ConnectServer* usa le impostazioni di *default* (dato che è senza parametri). L'ultima istruzione imposta il livello di sicurezza, questa, però, in *Windows 2000* e versioni successive è necessaria solo se dall'amministratore di sistema sono state modificate le impostazioni di sicurezza di *default*.

SCRIPTING CON WMI

In questo paragrafo presentiamo due *Script* uno *WSH* e uno *VBScript*. Il primo lo impostiamo in un file testo salvato con estensione *VBS*, l'altro lo definiamo dentro l'*IDE* di *Visual Basic*, così vediamo come utilizzare *WMI* senza referenziare la libreria *Wbemdisp*. Lo *Script WSH* permette di ricavare la quantità di memoria installata sul computer e quella disponibile nel momento in cui viene eseguito lo *Script*. In esso utilizziamo le classi *OperatingSystem* e *PhysicalMemory*, vediamo.

1 Le istruzioni per collegarsi alle due classi sono le seguenti.

```
Dim obj, SystemSet, MemorySet
Dim TotMem, FreeMem, Ram
Dim Services
Set Services = GetObject("winmgmts:")
Set SystemSet = Services.InstancesOf _
("Win32_OperatingSystem")
Set MemorySet = GetObject("winmgmts:"). _
InstancesOf("Win32_PhysicalMemory")
```

Facciamo notare che il collegamento alle classi è fatto con il metodo *InstancesOf*. Questo collegamento può essere fatto, anche, con un'unica istruzione, come nel caso di *Win32_PhysicalMemory*.

2 Le istruzioni per ricavare la memoria fisica installata e quella disponibile sono le seguenti.

```
For Each obj In SystemSet
 FreeMem = int(obj.FreePhysicalmemory)
Next
For Each obj In MemorySet
 Ram =int(obj.Capacity)
 TotMem=TotMem+Ram
Next
Wscript.Echo "-- … --" + chr(13) _
&"Ram Libera: " & FreeMem & " kb"+ chr(13)_
&"-- … --"+ chr(13)_
```

WMI QUERY LANGUAGE

WQL il linguaggio d'interrogazione di WMI è un subset dell'ANSI SQL con minore semantica e diverse limitazioni. Le query di WQL sono eseguite sulle classi e non sulle tabelle, come SQL, inoltre WQL non supporta query che coinvolgono più spazi dei nomi o query di aggiornamento o inserimento. Le principali parole chiavi del linguaggio sono le seguenti: SELECT, FROM, CLASS, LIKE, NOT, OR, GROUP CLAUSE, HAVING, IS e *ISA*. Per esempio la seguente query:

```
SELECT * FROM Win32_
LogicalDisk WHERE
FileSystem = "NTFS"
```

restituisce solo le istanze della classe *Win32_LogicalDisk* che rappresentano un drive *NTFS*.

```
&"Ram Totale: " & Round(TotMem/1024,2) & " Kb"
```

Fig. 3: *Il risultato dello Script WSH.*

Dato che dal metodo *InstancesOf* sono restituite delle collezioni di oggetti è necessario utilizzare due cicli *For Each*. In essi con la proprietà *FreePhysicalmemory* ricaviamo, in *Kb*, la memoria libera mentre con *Capacity* ricaviamo, in *Byte*, la capacità dei singoli banchi di memoria.

Per quanto riguarda l'esempio *VBScript* implementato dentro l'*IDE*, controllate i punti seguenti.

1 Create un nuovo progetto con una *form* sulla quale disponete un Pulsante (nominato *Sistema*) ed un *TextBox* multilinea. Le istruzioni da prevedere nel pulsante sono le seguenti.

```
Private Sub Sistema_Click()
 Dim SystemSet, System
 Dim INFO
 Set SystemSet = _
 GetObject("winmgmts:"). _
 InstancesOf("Win32_OperatingSystem")
 For Each System In SystemSet
  INFO = System.Caption & vbCrLf
  INFO = INFO & System.Manufacturer & vbCrLf
  INFO = INFO & System.BuildType & vbCrLf
  INFO = INFO & "Version: " + System.Version &
                                        vbCrLf
```

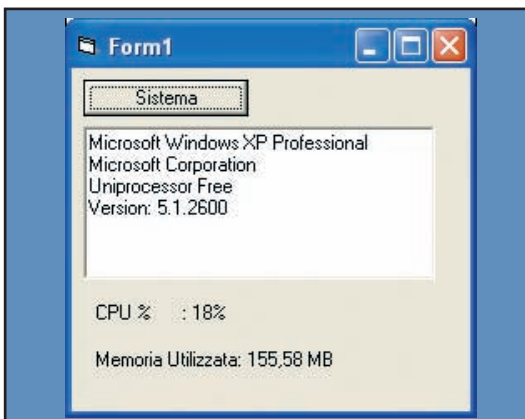



Fig. 4: *La form con i dati del sistema operativo*

```
Next
Text1 = INFO
End Sub
```

Anche in questo caso utilizziamo alcune proprietà della classe *OperatingSystem* e cioè: *Caption*, *Manufacturer*, *BuildType* e *Version*.

2 Questi due esempi possono essere adattati per creare un'applicazione che in tempo reale mostra la memoria disponibile, la quantità di *CPU* utilizzata e con opportuni controlli, fa scattare un allarme quando la memoria o la CPU superano determinate percentuali di utilizzo.

WMI E VISUAL BASIC

In questo paragrafo presentiamo un'applicazione *Visual Basic* che interagisce con alcune classi di *WMI* utilizzando la libreria *Wbemdisp*. L'applicazione permette di creare o terminare un processo, d'impostare la stampante predefinita, di avere informazioni sui dischi del sistema e di modificarne la descrizione. Inoltre, permette di ricavare informazioni: sulla scheda video (memoria e risoluzione video), sulla memoria centrale e sui punti di *Restore* dell'utility *Ripristino configurazione di sistema*. Per realizzare l'applicazione seguite i seguenti passi.

1 Create un nuovo progetto e tra i riferimenti inserite la libreria *MS WMI Scripting* (*Wbemdisp.tlb*), mentre tra i componenti referenziate: *MS Common Dialog Control 6.0* e *MS Windows Common Controls 6.0*. Questi due componenti contengono il Controllo *Common Dialog* e la *ListView*.

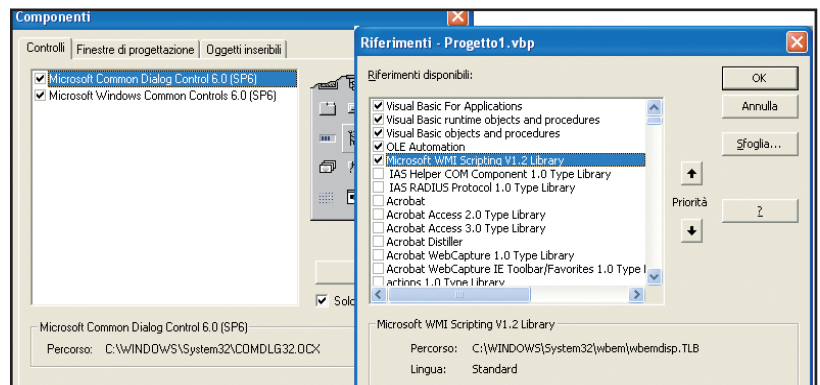
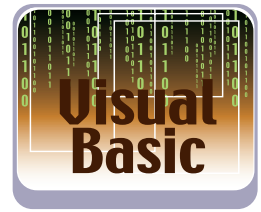



Fig. 5: *I riferimenti del progetto*

2 Sulla *Form* del progetto sistemate, anche con l'ausilio di qualche *Frame*: la *ListView*, un *ComboBox*, 9 pulsanti, una *ListBox* e un *CommonDialog*. Controllate la **Figura 6.**

WMI TOOLS

Per agevolare l'esplorazione delle classi *WMI* sul sito della Microsoft è disponibile *WMI TOOLS*. Esso include quattro strumenti: *WMI CIM Studio, WMI Object Browser, WMI Event Registration Tool* e *WMI Event Viewer*. Con questi strumenti è possibile esplorare il contenuto delle classi e del *repository*, eseguire dei metodi, visualizzare i valori delle proprietà ecc. Inoltre è possibile visualizzare e controllare i principali eventi.

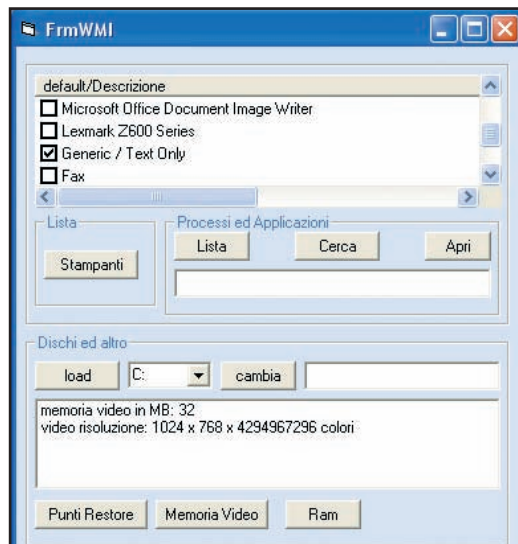
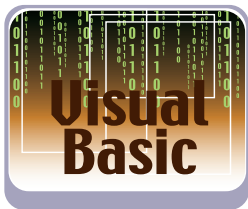



Fig. 6: *La form che interagisce con WMI*

3 Iniziamo presentando il codice che permette di mostrare, nella *ListView*: le stampanti istallate sul computer (pulsante *stampanti*) oppure l'elenco dei *processi* in esecuzione (pulsante *lista* del frame *Processi* ed *Applicazioni*). Nella parte dichiarativa inseriamo due variabili globali e nella *Form_Load* prevediamo il codice per impostare il *ListView1*.

```
Option Explicit
Dim Clickstampante As Boolean
    Dim ColItemStam As Object
Private Sub Form_Load()
 listview1.Appearance = ccFlat
 listview1.View = lvwReport
 listview1.Checkboxes = True
 listview1.FullRowSelect = True
End Sub
```

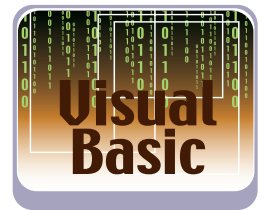
La variabile *Clickstampante* è impostata quando è premuto il pulsante stampa (ed è utilizzata nella procedura *listview1_ItemCheck)*, la *ColItemStam* invece è utilizzata come collezione di oggetti *Win32_Printer*. Nella *Form_Load* impostiamo la *ListView1*. Le procedure che permettono di elencare i processi in esecuzione sono le seguenti.

```
Private Sub elencoprocessi_Click()
 listview1.ColumnHeaders.Clear
 listview1.ColumnHeaders.Add , , "Descrizione", 3000
 listview1.ColumnHeaders.Add , , "Percorso di
                                   esecuzione", 5000
 listprocessi
End Sub
Private Sub listprocessi()
 Dim objProc As SWbemObject
 Dim strKey As String
 listview1.ListItems.Clear
 For Each objProc In ConnettiOggetti("WIN32_Process")
  strKey = objProc.Path_.Path
  listview1.ListItems.Add , strKey, objProc.Caption
  listview1.ListItems(strKey).SubItems(1) = _
  IIf(IsNull(objProc.ExecutablePath), "", _
  objProc.ExecutablePath)
 Next objProc
End Sub
Public Function ConnettiOggetti(strWMIClass _
As String) As SWbemObjectSet
 Dim objLocator As New SWbemLocator
 Dim objService As SWbemServices
 Set objService = objLocator.ConnectServer()
 Set ConnettiOggetti =
                   objService.InstancesOf(strWMIClass)
End Function
```

La *elencoprocessi* dopo aver impostato i titoli delle colonne della *ListView* invoca la procedura *listprocessi* che legge le istanze della classe *Process* ed inserisce nella *ListView* il *Path* (proprietà *Path_.Path*) dell'oggetto *SWbemObject* che rappresenta il processo e il *Path* di esecuzione del processo (*ExecutablePath*). Invece, la procedura che inserisce le descrizioni delle stampanti del sistema nella *ListView* è la seguente.

```
Private Sub Stampanti_Click()
 Dim objLocator As SWbemLocator
 Dim objWMIService As SWbemServices
 Clickstampante = True
 listview1.ColumnHeaders.Clear
 listview1.ColumnHeaders.Add , _
 , "default/Descrizione", 10000
 Set objLocator = New SWbemLocator
 Set objWMIService = objLocator.ConnectServer()
 listview1.ListItems.Clear
 Dim strKey As String
 Dim objitem As Object
 Set colItems = objWMIService.ExecQuery( _
 "Select * from Win32_Printer")
 For Each objitem In ColItemStam
  strKey = objitem.Caption
  listview1.ListItems.Add , strKey, strKey
```

SPAZIO DEI NOMI E CONTROLLO WMI

Uno *spazio dei nomi* è un insieme di nomi univoci utilizzato per individuare risorse ed elementi del sistema. Lo spazio dei nomi di default per *WMI* è *root \cimv2*. Questo per *Windows XP* e *Windows 2000* è inserito nella chiave di registro *HKEY_LOCAL_MACHINE\Software\Microsoft\WBEM \Scripting\Default Namespace*. Gli amministratori di sistema possono autorizzare singoli utenti o gruppi per l'accesso e l'esecuzione dei *Servizi WMI*, in particolare i livelli di autorizzazione possono essere messi in relazione agli spazi dei nomi. Questo aspetto lo potete verificare con il *Controllo WMI* di *Windows XP* che si trova nello Strumento di Amministrazione *Gestione Computer*.

```
If objitem.Default Then
 listview1.ListItems(strKey).Checked = True
End If
Next
End Sub
```

Nella procedura precedente gli elementi della classe *Printer* vengono ricavati attraverso la query *WQL "Select * from Win32_Printer"* (cioè seleziona tutte le istanze della classe *Printer*) e vengono inserite nella collezione *ColItemStam* (variabile globale). Notate che della classe *Win32_Printer* utilizziamo la proprietà *Caption* (nome della stampante) e *Default* che è un *Boolean* che stabilisce se la stampante selezionata è quella *predefinita*.

4 Quando sulla *ListView* sono mostrate le stampanti, quella *predefinita* è segnalata con il *CheckBox* spuntato. Quando, invece, sono mostrati i processi per chiuderne uno basta spuntare il relativo *CheckBox*. Il codice per gestire queste funzionalità è il seguente.

```
Private Sub listview1_ItemCheck(ByVal _
                              Item As MSComctlLib.ListItem)
 Dim DescrOggetto As String
 Dim objitem As Object
 Dim objProc As SWbemObject
 If Item.Checked Then
  If Clickstampante Then
   For Each objitem In colItems
    If objitem.Default = True Then
     listview1.ListItems(objitem.Caption) _
     .Checked = False
    End If
    If objitem.Caption = Item.Key Then
     objitem.SetDefaultPrinter
    End If
   Next
   Exit Sub
  End If
  If MsgBox("Vuoi disabilitare il processo?", _
   vbYesNo + vbQuestion) = vbYes Then
   DescrOggetto = Item.Key
   Set objProc = ConnettiOggetto(DescrOggetto)
   objProc.Terminate
   MsgBox "Processo Terminato"
   listview1.ListItems.Remove (DescrOggetto)
  End If
 End If

End Sub
```

Facciamo notare che per impostare la stampante predefinita utilizziamo il metodo *SetDefaultPrinter* della classe *Printer*, mentre per terminare un processo utilizziamo il metodo *Terminate* di *Process*. La funzione *ConnettiOggetto* la trovate nel *CD* allegato alla rivista.

5 Un'altra funzionalità relativa alla *ListView*, e ai pulsanti del *frame Processi ed Applicazioni*, è la ricerca di un'applicazione (file *EXE*) e la creazione del relativo processo di esecuzione. A tal file per ricercare l'applicazione è utilizzato il pulsante *Cerca*, il *TextBox TxtApplicazione* e naturalmente il *CommanDialog*.

6 Infine il codice per creare il processo di esecuzione di un'applicazione precedentemente scelta è il seguente

```
Private Sub nuovop_Click()
 If TxtApplicazione <> "" Then
  If Not NuovoProcesso(TxtApplicazione) Then
    MsgBox "Non è possibile creare il processo"
   Else
   listprocessi
  End If
  TxtApplicazione = ""
 End If
End Sub

Function NuovoProcesso(applicazione _
                                    As String) As Boolean
 Dim IDProcess As Long
 Dim objProcess As SWbemObject
 Dim objStartup As SWbemObject
 Dim objconfig As SWbemObject
 Set objStartup =
                 ConnettiOggetto("Win32_ProcessStartup")
 Set objProcess =
                         ConnettiOggetto("Win32_Process")
 Set objconfig = objStartup.SpawnInstance_
 objconfig.ShowWindow = 1
 NuovoProcesso =
                     IIf(objProcess.Create(applicazione, _
               Null, objconfig, IDProcess) = 0, True, False)
 Exit Function
errore:
 NuovoProcesso = False

End Function
```

Nel CD allegato alla rivista trovate il progetto completo.

CONCLUSIONE

L'articolo è un valido ausilio per chi vuole iniziare ad esplorare il *WMI* e le funzionalità del sistema operativo. Gli esempi presentati sono interessanti ed illustrano solo alcune delle potenzialità del *WMI*.

Massimo Autiero

Manipolare le immagini in Java

Con il successo dei photoblog nasce il problema di ridimensionare le fotografie. Realizziamo un programma per visualizzare le immagini in una directory, selezionarle e ridimensionarle tutte con un clic

In questo articolo verrà illustrato come realizzare un semplice programma di ridimensionamento di immagini. Queste possono essere ad esempio fotografie scattate con la vostra nuova macchina fotografica digitale. L'idea è quella di ottenere un software che "Scansioni" una directory alla caccia di immagini JPEG, GIG, PNG. Generi un'anteprima nell'interfaccia del programma, consenta di selezionare quelle da ridimensionare e infine le ridimensioni.
L'interfaccia del nostro programma sarà molto semplice e riportata in **Figura 1.**

Fig. 1: **L'interfaccia utente di MyImages**

FUNZIONAMENTO DEL PROGRAMMA

Nell'interfaccia si nota la presenza di un campo di testo affiancato da due pulsanti. In questo campo è necessario digitare il percorso da dove prelevare le immagini. Il pulsante *Cerca* invece avvia la scansione della directory selezionata. Durante questa fase vengono individuati i file del formato supportato e ne viene creata una piccola anteprima *(thumbnail)*, che verrà utilizzata per la selezione delle foto su cui operare. Durante la scansione viene visualizzato il nome del file ed aggiornata una barra di progressione. Una volta terminata la ricerca, il programma ridimensiona la finestra principale, facendo apparire un elenco. In questo elenco sono presenti le anteprime ed i nomi dei file. È possibile selezionare le fotografie da ridimensionare e fare clic sul pulsante Esporta. Se non si seleziona alcuna fotografia vengono esportate tutte le foto.
Vengono poi create due directory: *preview* e *data*. La prima contiene una anteprima, mentre la seconda la foto ridimensionata.

DOVE TUTTO INIZIA

La classe principale del progetto è *Main*, il cui codice sorgente è il seguente:

```
package it.bigatti.myimages;
public class Main {
    public static void main(String[] args) {
        PreviewFrame f = new PreviewFrame();
        f.setLocation(25, 25);
        f.setVisible(true); }
}
```

come si vede viene creata e visualizzata una finestra *PreviewFrame*, che è quella che implementa tutta l'interfaccia utente del programma.
La classe *PreviewFrame* sarà derivata da JFrame, in quanto si desidera realizzare una finestra Swing.

```
public class PreviewFrame extends JFrame {
```

All'interno della classe saranno presenti i campi, necessari a gestire l'interazione con l'utente.

```
        JTextField pathTextField;
        JList imageList;
        JProgressBar progressBar;
        JLabel progressText;
        JPanel imagesPanel;
```

```
        JButton browseButton;
        JButton findButton;
        JButton exportButton;
```

Il costruttore della classe non fa altro che richiamare la classe base impostando il titolo *"myImages"*. Viene poi creato un oggetto *ImageManager*, che verrà utilizzato per le operazioni di caricamento ed esportazione immagini. La creazione dell'interfaccia utente è assegnata al metodo *createUI()*:

```
        public PreviewFrame() {
          super("myImages");
          manager = new ImageManager();
          createUI();
        }
```

il metodo *createUI()* è il seguente:

```
void createUI() {
    Container c = getContentPane();
    c.setLayout( new BorderLayout() );
    c.add( createTop(), BorderLayout.NORTH );
    c.add( createContents(), BorderLayout.CENTER );
    c.add( createBottom(), BorderLayout.SOUTH );
   pack();
}
```

come si vede, viene ottenuto il contenitore dei componenti visuali della finestra. A questo viene assegnato un Layout Manager di tipo *BorderLayout*. Questo dispone i componenti nelle quattro posizioni cardinali, più uno al centro. Ne verrà fatto largo uso anche in seguito. Poi vengono create le tre zone principali:

- in alto viene posizionato un pannello che comprende il campo di testo e i due pulsanti;
- al centro è presente l'elenco di selezione ed il pulsante di esportazione;
- in basso viene posizionata la barra di stato, composta da una barra di progressione ed una etichetta di testo.

COSTRUIRE L'INTERFACCIA UTENTE

Non trascrivo il codice intero per la gestione dell'interfaccia utente, che comunque potete trovare nel cd allegato, mentre invece riporto solo alcuni elementi essenziali, che serviranno alla comprensione del resto dell'articolo.
Nel pannello superiore troviamo:

```
Container createTop() {
  [...]
  pathTextField = new JTextField(25);
  pathTextField.setText( prefs.getPath() );
  [...]
  findButton = new JButton("Cerca");
  findButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
      setLock(true);
      loadThumbnails(); } } );
  [...]
  return p;
}
```

Si noti la presenza di un gestore di evento sul pulsante di ricerca. Questo invoca i metodi:

- **setLock()**, che blocca l'interfaccia utente per impedire all'utente di operare con il programma mentre questo sta già facendo qualcosa;
- **loadThumbnails()**, che carica le anteprime delle immagini.

Il pannello principale è costituito da un elenco implementato da una JList. A questo è stato aggiunto un *ListSelectionListener*, che invoca il metodo *updateSelection()*. Questo visualizza nella barra di stato il numero di elementi selezionati.
Alla *JList* viene anche associato un *CellRenderer* di tipo *ImageInfoCellRenderer*, che si occupa di visualizzare, oltre al nome del file, anche l'immagine di anteprima:

```
Container createContents() {
  [...]
  imageList = new JList();
  imageList.addListSelectionListener(
  new ListSelectionListener() {
    public void valueChanged(ListSelectionEvent ev) {
      if (!ev.getValueIsAdjusting()) {updateSelection();
    } }
});
imageList.setCellRenderer( new
                        ImageInfoCellRenderer() );
  [...]
  exportButton = new JButton("Esporta");
  exportButton.addActionListener(new
                                  ActionListener() {
    public void actionPerformed(ActionEvent ev) {
      setLock(true);
      export(); }
  } );
  [...]
  return imagesPanel;
}
```

Il pulsante di esportazione invece invoca il metodo *export()*. Il pannello conclusivo invece è molto più semplice, perché composto solo dalla barra di progressione e da una etichetta di stato.

DOVE SONO GLI IMPORT?

Gli import e gli elementi secondari sono disponibili nel codice allegato all'articolo. Qui per ragioni di spazio riporteremo solo i punti essenziali

COME INIZIARE

Questo progetto richiede JAI. Ulteriori informazioni su Java Advanced Imaging sono reperibili in Internet sul sito http://java.sun.com/products/java-media/jai/ dove è possibile accedere ai download delle librerie, di codice aggiuntivo, della documentazione ed accedere a link di articoli correlati alla tecnologia. Sono presenti anche dei business case sui clienti che hanno adottato JAI.

CARICARE LE ANTEPRIME

Il caricamento delle anteprime delle fotografie è affidato al metodo *loadThumbnails()*, che utilizza un thread separato. Questo è necessario per permettere al thread principale di esecuzione di continuare ad aggiornare e gestire l'interfaccia utente del programma. Se non si fosse fatto in questo modo, non si sarebbe visto l'avanzamento dello stato durante il caricamento delle foto. Il metodo per prima cosa recupera il percorso dal campo di testo e lo imposta sull'oggetto Preferences, poi chiede ad *ImageManager* di caricare le immagini richieste, le scorre, e per ciascuna chiama il metodo *loadThumbnail()*. Nel fare questo aggiorna la barra di stato. Al termine del caricamento:

- associa l'elenco delle immagini caricate alla JList;
- ridimensiona la finestra;
- sblocca l'interfaccia utente.

Il codice è il seguente:

```
private void loadThumbnails() {
   prefs.setPath( pathTextField.getText() );
   Thread t = new Thread( new Runnable() {
      public void run() {
         images = manager.loadImages(
                              pathTextField.getText() );
         progressBar.setMinimum( 0 );
         progressBar.setMaximum( images.size()-1 );
         for (int i=0; i<images.size(); i++) {
            ImageInfo info = (ImageInfo)images.get(i);
            progressText.setText( info.getFilename() );
            progressBar.setValue( i );
            info.loadThumbnail();
            progressBar.setValue(0);
            progressText.setText( "" );
            imageList.setListData(images.toArray(
                        new ImageInfo[ images.size() ]));
            imagesPanel.setVisible(true);
            Dimension d = getSize();
            setSize((int) d.getWidth(), 500);
            setLock(false); } });
   t.start();
}
```

NOTA

I PANNELLI

Nel costruire l'interfaccia grafica del progetto spesso sono stati utilizzati pannelli annidati l'uno dentro l'altro. Questo approccio si rende indispensabile in quanto in Java la gestione della posizione e dimensione dei componenti visuali è gestita da oggetti specifici, chiamati Layout Manager. Questi oggetti distribuiscono spazio e posizionano ciascuno in modo diverso. Utilizzando quelli più semplici è però necessario, per ottenere una interfaccia utente non banale, utilizzarne diversi, in pannelli annidati.

GESTIRE LE IMMAGINI

La classe *ImageManager* si occupa, tra l'altro, di caricare le immagini dalla directory indicata, per fare questo, utilizza la classe *File*, fornendo un *FilenameFilter* che accetta solo i file supportati. Questi sono determinati dal metodo *isSupported()*, che non fa altro che controllare che l'estensione del file sia una di quelle elencate:

```
[...]
public class ImageManager {
   String[] formatiSupportati = { ".jpg", ".jpeg",
                                        ".gif", ".png" };
   public List loadImages( String path ) {
      List result = new ArrayList();
      File pathFile = new File(path);
      File[] elenco = pathFile.listFiles(
                              new FilenameFilter() {
         public boolean accept(File dir, String file) {
            return isSupported(file); } } );
      if (elenco != null) {
         for( int i=0; i<elenco.length; i++ ) {
            result.add( new ImageInfo( elenco[i] ));
      } }
   return result;
}
```

Il metodo *isSupported()* non fa altro che verificare che il nome di file fornito termini per una delle estensioni supportate. Nel fare il confronto, il nome viene trasformato in sole lettere minuscole, per intercettare anche nomi di file che utilizzano lettere maiuscole. Il metodo *endsWith()* utilizzato, infatti, è sensibile alle maiuscole/minuscole:

```
boolean isSupported( String filename ) {
   for( int i=0; i<formatiSupportati.length; i++ ) {
      if (filename.toLowerCase().endsWith(
                              formatiSupportati[i])) {
         return true; } }
   return false;
}
```

Il risultato è una lista di oggetti *ImageInfo* di cui si è ragionevolmente sicuri puntino a file di immagine. Il controllo è stato eseguito solo sull'estensione. D'altra parte, verifiche più approfondite, a questo punto, sarebbero forse eccessive.

INFORMAZIONI SULLE IMMAGINI

La classe *ImageInfo* rappresenta un'immagine e fornisce dei metodi per ottenerne informazioni. In particolare è possibile ottenere:

- una anteprima;
- una anteprima quando l'icona è selezionata;
- il nome del file.

La classe contiene alcuni campi:

- l'oggetto *File* che rappresenta il file immagine;
- un oggetto *PlanarImage* che contiene l'immagine caricata in memoria;
- due icone per le anteprime;

- la larghezza standard delle anteprime.

La classe è così dichiarata:

```
public class ImageInfo {
        File imageFile;
        PlanarImage image;
        Icon thumbnailIcon;
        Icon darkThumbnailIcon;
        final int thumbnailWidth = 75;
        public ImageInfo( File imageFile ) {
                this.imageFile = imageFile;
        }
```

Il metodo *loadThumbnail()* si occupa di creare le due anteprime. Il suo funzionamento è il seguente:

- carica l'immagine con il metodo *load();*
- determina una scala di ridimensionamento per portare l'immagine, la cui dimensione può variare, alla larghezza standard dell'anteprima;
- crea due oggetti *MyIcon* con l'immagine ridimensionata;
- rilascia l'immagine dalla memoria.

Questo ultimo passaggio è fondamentale perché è impossibile mantenere in memoria più di poche immagini in formato pieno. Sia per la fase di creazione delle anteprime che per quella successiva di esportazione è dunque necessario operare con una immagine alla volta, caricandola, elaborandola ed infine rilasciandola dalla memoria:

```
public void loadThumbnail() {
  load();
  float scale = getThumbnailScale(thumbnailWidth);
  System.out.println("ImageInfo.getThumbnailIcon():
        creazione " + imageFile + " scala " + scale);
  thumbnailIcon = new MyIcon( getScaledImage(
                                   scale ), false );
  darkThumbnailIcon = new MyIcon(
                       getScaledImage( scale ), true );
  System.out.println("ImageInfo.getThumbnailIcon():
                                           creato");
  release();
}
```

Il metodo *load()* utilizza le API JAI *(Java Advanced Imaging)* per caricare l'immagine in un oggetto *PlanarImage:*

```
  public void load() {
    System.out.println("ImageInfo.load():
                       caricamento " + imageFile);
    image = JAI.create("fileload",
                       imageFile.getAbsolutePath());
  }
```

La fase di rilascio della memoria, richiede solo l'assegnazione del riferimento all'immagine di un valore *null*. Il resto delle operazioni è svolto dal garbage collector: non appena sarà necessario caricare una nuova immagine, sarà quest'ultimo a scaricare effettivamente dalla memoria l'immagine precedente:

```
public void release() {
  image = null;
  System.out.println("ImageInfo.release(): memoria
                                       rilasciata");
}
```

Per determinare la scala di ridimensionamento viene utilizzato questo metodo, che non fa altro che ottenere dall'immagine l'altezza e la larghezza. Poi determina se è un ritratto o un paesaggio e calcola le proporzioni di conseguenza:

```
 public float getThumbnailScale(int requiredWidth) {
    int height = image.getHeight();
    int width = image.getWidth();
    boolean tall = (height > width);
    return requiredWidth / (float) (tall ? height :
                                           width);
 }
```

Il valore *float* risultante è il valore di scala. È possibile eseguire i calcoli anche con valori double, ma in questo caso sono stati riscontrati dei malfunzionamenti in fase di ridimensionamento.

Il metodo *getScaledImage()* è quello che si occupa di eseguire il ridimensionamento dell'immagine, utilizzando le chiamate JAI. Per prima cosa viene creato un oggetto *ParameterBlock*, che ospiterà tutti i parametri. Questi sono:

- l'immagine da ridimensionare;
- il fattore di ridimensionamento orizzontale;
- il fattore di ridimensionamento verticale;
- eventuale traslazione orizzontale;
- eventuale traslazione verticale;
- il metodo di interpolazione richiesto.

Poi viene chiamato JAI richiedendo un'operazione di ridimensionamento. Questo ritorna una immagine *PlanarImage*, che viene convertita immediatamente ad una immagine *BufferedImage* e ritornata. Il codice è il seguente:

```
public BufferedImage getScaledImage(float scale) {
  ParameterBlock params = new ParameterBlock();
  params.addSource(image);
  params.add(scale);//x scale factor
  params.add(scale);//y scale factor
  params.add(0.0F);//x translate
  params.add(0.0F);//y translate
```

NOTA

CONTROLLO DELLA POSIZIONE

Oltre ai Layout Manager, in Java è possibile utilizzare il posizionamento assoluto. È sufficiente impostare il manager a null utilizzando il metodo setLayout() su un qualsiasi componente visuale che sia anche un contenitore (sottoclasse di Container). In questo modo è possibile controllare con precisione la posizione e la dimensione dei componenti.
Il posizionamento fisso, senza Layout Manager, permette di ottenere un completo controllo sulla posizione e dimensione dei componenti presenti in un contenitore. Purtroppo, in questo caso, è necessario gestire tutte le eventuali operazioni di ridimensionamento e riposizionamento, che spesso possono diventare molto lunghe e complesse. Decisamente più complesse che utilizzare Layout Manager e contenitori annidati.

```
params.add(new InterpolationNearest());
                            //interpolation method
PlanarImage scaled = JAI.create("scale", params);
return scaled.getAsBufferedImage();
}
```

Il cambio di tipo di immagine è necessario in quanto le operazioni che verranno svolte in seguito richiedono un oggetto che derivi da Image, mentre *PlanarImage* non lo è. Questa classe infatti è specifica di JAI e non si basa sul concetto di immagine presente nel resto della piattaforma Java. Fortunatamente, esiste il metodo *getAsBufferedImage()* che funge da "ponte" tra JAI ed il resto delle API di Java.
È necessario anche sottolineare che il tipo di interpolazione scelto incide sulla qualità finale. Oltre ad *InterpolationNearest* sono disponibili anche:

- *InterpolationBicubic;*
- *InterpolationBicubic2;*
- *InterpolationBilinear;*
- *InterpolationTable;*

Ciascuna di queste classi implementa un algoritmo di interpolazione diverso. Alcuni sono più precisi, ma richiedono anche più tempo di elaborazione. Un esempio è l'interpolazione bicubica. In questo progetto è stata sempre utilizzata l'interpolazione Nearest, che non produce il miglior risultato possibile. Può andar bene per le anteprime che si vedono nel programma, ma in fase di esportazione potrebbe essere meglio utilizzare un algoritmo più lento ma che produca un risultato di maggiore qualità. La classe si conclude con alcuni metodi di supporto, che non sono altro che getter per specifiche proprietà:

NOTA

LA CLASSE PREFERENCES

La classe Preferences è un Singleton. Questo nome identifica un design pattern e vuol dire che in tutta l'applicazione è presente una sola istanza di questa classe. Si è scelto di fare in questo modo per avere un punto unico di accesso alla classe che gestisce le preferenze e perché sia il codice stesso ad assicurarsi che il programmatore, neanche per errore, possa creare più oggetti Preferences.

IMMAGINI ED ICONE

La classe *ImageInfo* include anche una classe interna, che implementa una specifica tipologia di icona. È una sottoclasse di *ImageIcon,* una classe di Swing che implementa una icona a partire da un oggetto *Image:*

```
class MyIcon extends ImageIcon {
    boolean isDark;
```

il costruttore si aspetta in input l'immagine da utilizzare ed un flag che indica se l'icona è selezionata o meno:

```
public MyIcon( Image image, boolean isDark ) {
    super(image);
    this.isDark = isDark;
}
```

il metodo che si occupa di disegnare l'icona è *paintIcon(),* che si aspetta in input il componente su cui operare, il contesto grafico su cui scrivere ed una coppia di coordinate. La nostra implementazione è semplice: per prima cosa viene determinata la dimensione dell'immagine e se questa è orizzontale o verticale. Poi vengono aggiustate le coordinate per fare in modo che, indipendentemente dall'orientamento dell'immagine, questa risulti centrata nello spazio disponibile. Poi l'icona viene disegnata con il metodo *drawImage().*
Al termine, se il flag *isDark* è stato settato, viene tracciato un riquadro di selezione attorno all'immagine:

```
public synchronized void paintIcon(
                  Component c, Graphics g, int x, int y) {
   int height = getImage().getHeight(
                                  getImageObserver());
int width = getImage().getWidth(
                                  getImageObserver());
boolean tall = (height > width);
if (tall) {
   x += (thumbnailWidth-width) / 2;
} else {
  y += (thumbnailWidth-height) / 2;}
g.drawImage( getImage(), x, y, getImageObserver());
if (isDark) {
   g.setColor( Color.BLUE );
   g.drawRoundRect(10, 10, thumbnailWidth,
                              thumbnailWidth, 10, 10);}
}
```

La classe contiene anche la ridefinizione di un paio di metodi, quelli relativi alla dimensione dell'icona. La classe ImageIcon ritornerebbe infatti la dimensione dell'immagine. Questa può variare in funzione dell'orientamento della foto. Nel nostro caso è quindi necessario ritornare un valore fisso, rappresentato dalla dimensione del thumbnail:

```
public int getIconHeight() {
   return thumbnailWidth; }
public int getIconWidth() {
   return thumbnailWidth; } }
}
```

DISEGNARE LE ANTEPRIME

Una volta visto come il sistema rappresenta una immagine, è possibile passare alla realizzazione della classe che si occupa della sua visualizzazione. Come si ricorderà dalla classe *PreviewFrame,* l'oggetto *JList* utilizzato per rappresentare l'elenco delle anteprime utilizza un *CellRenderer* personalizzato: *ImageInfoCellRenderer.*

Questa è una sottoclasse di JLabel che ha la caratteristica di visualizzare una icona insieme a del testo. La classe implementa l'interfaccia *ListCellRenderer:*

```
public class ImageInfoCellRenderer extends JLabel
                    implements ListCellRenderer {
```

viene definito un colore, utilizzato per evidenziare le righe selezionate:

```
  private static final Color SELECTED_COLOR = new
                                Color(61,128,223);
```

il costruttore non fa altro che impostare il flag di opacità a *true* ed impostare un bordo vuoto per distanziare ciascuna riga dell'elenco.

```
   public ImageInfoCellRenderer() {
       setOpaque(true);
       setBorder( BorderFactory.createEmptyBorder(
                                   10, 10, 10, 10 ));
  }
```

Il componente di visualizzazione degli elementi nella lista viene ritornato dal metodo *getListCellRendererComponent()*. In questo caso viene ritornato l'oggetto stesso, e cioè *this*. Prima però vengono impostati tutti gli attributi necessari. Per prima cosa il parametro value, che rappresenta il valore dell'elemento corrente, viene convertito in un *ImageInfo*. La lista infatti contiene solo elementi di questo tipo. L'oggetto *ImageInfo* contiene tutte le informazioni necessarie alla visualizzazione, tra cui il testo e l'icona. Quest'ultima viene ottenuta in modo diverso in funzione del flag *isSelected*. Se la riga è selezionata, viene utilizzata l'icona *"dark"*, altrimenti quella normale. In seguito vengono impostati i colori di primo piano e sfondo, in funzione dello stato di selezione della riga. Il fatto che il flag di opacità sia stato impostato a true prevede che il colore di sfondo sia utilizzato per il riempimento dell'area del componente. Se fosse stato *false*, lo sfondo non sarebbe stato disegnato:

```
public Component getListCellRendererComponent(
   JList list, Object value, int index, boolean isSelected,
                               boolean cellHasFocus) {
  ImageInfo info = (ImageInfo)value;
  setText(info.getFilename());
  if (isSelected) {
      setIcon(info.getDarkThumbnailIcon());
  } else {
      setIcon(info.getThumbnailIcon());}
      setBackground(isSelected ? SELECTED_COLOR :
                                      Color.white);
      setForeground(isSelected ? Color.white :
                                      Color.black);
      return this; }
}
```

ESPORTAZIONE E FILE INDICE

Alla pressione del pulsante *Esporta* viene prodotto il file indice delle immagini selezionate e vengono esportate due versioni di dimensione ridotta: l'anteprima e l'immagine ridimensionata. Queste operazioni vengono coordinate dal metodo *export()*, che viene chiamato alla pressione del pulsante. La prima operazione che viene svolta da questo metodo è quella di selezionare tutte le immagini se l'utente non ne ha selezionata alcuna:

```
void export() {
  if (imageList.getSelectedIndices().length == 0) {
    imageList.addSelectionInterval(0, images.size()-1);
  }
```

poi viene creato un nuovo thread. Come per l'operazione di creazione, non si vuole bloccare l'interfaccia utente, durante l'intensiva operazione di esportazione dei dati:

```
Thread t = new Thread( new Runnable() {
   public void run() {
      try {
```

nel thread viene estratto l'elenco delle immagini selezionate, sottoforma di indici. Viene creata una lista di elementi selezionati ed inizializzata la barra di progressione. Poi viene eseguito un ciclo per ogni immagine selezionata e su ciascuna di queste viene chiamato il metodo *export()* sulla classe *ImageManager.* Sarà questo metodo ad eseguire la vera e propria operazione di esportazione:

```
int[] selected = imageList.getSelectedIndices();
List selectedList = new ArrayList();
progressBar.setMaximum(selected.length-1);
   for( int i=0; i<selected.length; i++) {
       ImageInfo info = (ImageInfo)images.get(
                                       selected[i]);
       progressBar.setValue(i);
       progressText.setText(info.getFilename());
       manager.export(pathTextField.getText(), info );
       selectedList.add( info );
   }
```

alla fine delle operazioni la barra di stato viene reimpostata e viene esportato il file indice, utilizzando il metodo *exportManifest()* della classe *ImageManager:*

```
   progressBar.setValue(0);
```

DIFFERENZE TRA IMMAGINI

Siccome Java supporta due nozioni che possono ad una prima occhiata sembrare simili: immagini ed icone. Non si deve però confondere Image con Icon. La prima è intesa come immagine ad ampio raggio, utilizzata dalle Applet, alle estensioni JAI, fino a Java2D ed all'accelerazione hardware. L'icona è invece un elemento grafico, solitamente di dimensioni più ridotte, utilizzato da Swing per illustrare le interfacce utente. Il ponte tra questi due mondi è ImageIcon.

```
    progressText.setText("");
    manager.exportManifest(
                  pathTextField.getText(),selectedList );
} catch( IOException e ) {
    progressText.setText( e.getMessage() );
    }
    setLock(false);
}
});
t.start();
}
```

Il metodo appena visto si occupa del coordinamento delle operazioni, ma è il metodo *export()* della classe *ImageManager* a fare tutto il lavoro:

```
public class ImageManager {
    //...
    public void export( String path, ImageInfo info )
    throws IOException {
```

Vengono creati due oggetti *File,* ciascuno dei quali rappresenta una directory di output. Per convenzione sono sottodirectory del percorso da cui sono state caricate le immagini:

```
File dataDir = new File( path + File.separator +
                          "data" + File.separator);
File previewDir = new File( path + File.separator +
                          "preview" + File.separator);
```

poi viene caricata l'immagine:

```
info.load();
```

viene tentata la creazione delle directory di output, nel caso non esistessero già:

```
if (!dataDir.mkdirs()) {
    if (!dataDir.canWrite()) {
        info.release();
        throw new IOException("Impossibile creare il
                      percorso " + dataDir.getName());
    }
}
```

viene poi determinata la scala di ridimensionamento da effettuare, utilizzando le preferenze memorizzate nella classe *Preferences.* Viene poi creato un oggetto *File* che rappresenta il file di output da generare. Questo viene passato poi al metodo *write()*, che produce il file, in formato jpeg. Questo metodo utilizza un grado di compressione di default:

```
float scale = info.getThumbnailScale(
        Preferences.getInstance().getDataWidth() );
File f = new File( dataDir.getAbsolutePath() +
                   File.separator + info.getFilename() );
ImageIO.write(info.getScaledImage(scale), "jpeg", f);
```

La stessa sequenza di operazioni utilizzata per produrre il file di output principale viene applicata per le anteprime, ovviamente utilizzando una dimensione di riferimento inferiore:

```
if (previewDir.mkdirs()) {
  if (!previewDir.canWrite()) {
    info.release();
    throw new IOException("Impossibile creare il p
             ercorso " + previewDir.getName());}
  }
  scale = info.getThumbnailScale(
      Preferences.getInstance().getPreviewWidth() );
  f = new File( previewDir.getAbsolutePath() +
  File.separator + info.getFilename() );
  ImageIO.write(info.getScaledImage(scale), "jpeg", f);
  info.release();
}
```

Il metodo di creazione del file indice è molto semplice, in quanto utilizza un oggetto *FileWriter* per creare un file *index.txt* nella directory dove sono state caricate le immagini. Il contenuto del file è semplicemente un elenco dei nomi di file contenuti nell'elenco delle immagini selezionate per l'esportazione:

```
public void exportManifest(String path, List
                                     selectedList)
throws IOException {
  FileWriter writer = new FileWriter(path +
                   File.separator + "index.txt");
  for( Iterator iter = selectedList.iterator();
                            iter.hasNext(); ) {
     ImageInfo info = (ImageInfo)iter.next();
     writer.write( info.getFilename() + ";\n");
   }
   writer.close();
 }

}
```

CONCLUSIONI

In questo progetto si è visto come realizzare un programma per l'esportazione di immagini, e relative anteprime, utilizzando Java e le API JAI. Sono stati illustrati anche elementi di interfaccia utente avanzata, come la personalizzazione della visualizzazione degli elementi all'interno di una JList. L'utilizzo dei thread ha permesso inoltre di eseguire le operazioni richieste fornendo contemporaneamente un feed-back all'utente.

Massimiliano Bigatti

NOTA

AWT O JAI?

Il ridimensionamento delle immagini in Java può essere fatto in diversi modi: utilizzando AWT, JAI ed in un paio di altri modi. Le prestazioni però cambiano da metodo a metodo, e così la qualità del risultato. La modalità utilizzata nell'articolo è un buon compromesso tra velocità di esecuzione, qualità e facilità di programmazione.

J2ME e la Push Technology

Una delle caratteristiche più interessanti, introdotte con MIDP 2.0, è il cosiddetto Push Registry. Tramite esso, infatti, una MIDlet può essere attivata da un evento esterno, come un timer programmato

Il linguaggio Java orientato ai dispositivi mobili, ovvero J2ME, è senza dubbio uno strumento molto potente per creare applicazioni destinate a cellulari, palmari e quant'altro. Con l'introduzione di MIDP 2.0 sono stati fornite ulteriori caratteristiche che concorrono a rendere il linguaggio della Sun ancora più potente. Tra queste caratteristiche, una molto interessante è il cosiddetto push registry.
La tecnologia push è concetto che va al di là di J2ME e MIDP 2.0. In generale, per push si intende quel meccanismo che permette di ricevere le informazioni in modo asincrono. Cioè, l'applicazione viene "svegliata" da un evento esterno e non dall'utente.
In MIDP 2.0 tale meccanismo è detto push registry e permette di attivare una MIDlet tramite un evento esterno. Vediamo in che modo una MIDlet può essere attivata.

ATTIVAZIONE DI UNA MIDLET

Prima dell'avvento di MIDP 2.0 una MIDlet poteva essere attivata solamente dall'utente, ovvero manualmente. Il push registry, invece, permette di attivare una MIDlet tramite un evento esterno. Vi sono due tipi di eventi che possono attivare una MIDlet e cioè tramite connessione di rete o tramite un timer programmato. Quindi, in definitiva, in MIDP 2.0 una MIDlet può essere attivata in tre modi:

- Richiesta dell'utente
- Connessione di rete
- Timer programmato

Ricordiamo, però, che dietro le quinte c'è sempre l'AMS (Application Management System). L'AMS è il software responsabile per la gestione del ciclo di vita di ciascuna MIDlet; quindi, anche se l'applicazione viene risvegliata da un evento esterno, è sempre l'AMS a gestirne lo stato.
Vediamo, ora, che tipi di connessioni di rete sono supportati. In MIDP 1.0 l'unico tipo di connessione supportata era HttpConnection. Oltre a questa, in MIDP 2.0 ne sono supportate altre due: connection-oriented e connectionless. Il primo tipo di connessione è supportato tramite i socket, che utilizzano il protocollo TCP. Il secondo tipo utilizza il protocollo UDP e i datagrammi. Per chi non è un esperto di reti non deve preoccuparsi. Vedremo che Java ci permette di prescindere dal dettaglio tecnico di questo tipo di connessioni.
A parte queste, vi sono le WMA (Wireless Messaging API) che permettono di attivare una MIDlet al verificarsi dell'arrivo di un messaggio, come un SMS. Le specifiche MIDP 2.0, però, non obbligano i costruttori dei dispositivi mobili ad implementare tutte le connessioni esposte. Generalmente, saranno disponibili i tipi socket e datagramma ma non è detto che, ad esempio, sia supportato WMA. In definitiva, abbiamo tre tipi di connessioni di rete che possono richiedere l'attivazione di una MIDlet:

- Message-based (SMS)
- Stream-based (socket)
- Packet-based (datagramma)

Come abbiamo già detto, vi è un altro tipo di evento che può risvegliare una MIDlet: un timer programmato. Ciò può essere molto

utile per ripetere un determinato evento sistematicamente. Potremmo, ad esempio, scrivere un'applicazione che funga da remainder per i nostri appuntamenti, oppure un'applicazione che interroghi un server, tramite una connessione di tipo HttpConnection, per vedere se ci sono aggiornamenti di qualche tipo.
A questo punto qualcuno potrebbe giustamente chiedersi: "ok ma, sostanzialmente, come possiamo comunicare all'AMS di svegliare l'applicazione da noi sviluppata al verificarsi di uno degli eventi visti?". Il prossimo paragrafo risponderà proprio a questo tipo di domanda.

REGISTRAZIONE DI UNA MIDLET

Una MIDlet deve comunicare, in qualche modo, all'AMS la sua intenzione di essere attivata al verificarsi di un particolare evento.
Questa comunicazione che la MIDlet fa all'AMS è detta registrazione.
Vi sono due tipi di registrazione che è possibile utilizzare:

- Registrazione statica.
- Registrazione dinamica.

Nel primo caso dobbiamo modificare il file JAD (Java Application Descriptor) associato alla MIDlet che stiamo sviluppando. La proprietà del file JAD che ci permette di definire una registrazione statica è MIDlet-Push-<n>, la sua struttura è la seguente:

```
MIDlet-Push-<n>: <URLDiConnessione>,
                        <NomeMIDlet>, <Restrizioni>
```

dove:

- **MIDlet-Push-<n>** è la proprietà che identifica una particolare registrazione. Il numero <n> è progressivo e parte da 1. Questo perché possiamo avere più di una registrazione per la stessa MIDlet. Ad esempio, una MIDlet potrebbe essere risvegliata sia da una connessione di tipo socket che da una di tipo datagramma. Un esempio è MIDlet-Push-1.

- **<URLDiConnessione>** rappresenta una stringa di connessione nella quale viene specificato sia il tipo di connessione che la porta. Ad esempio, socket://:8000.

- **<NomeMIDlet>** è il nome della classe rappresentante la MIDlet che dovrà essere svegliata al verificarsi dell'evento esterno. In pratica, appena si verifica l'evento viene invocato il metodo startApp relativo alla MIDlet indicata. Ad esempio, <NomeMIDlet> potrebbe essere rappresentata da *j2meprojects.pushtechnology.PushMIDlet.*

- **<Restrizioni>** non è altro che un filtro utilizzato per specificare chi può richiedere il risveglio della MIDlet. Vi sono casi in cui, infatti, vorremmo permettere il push delle informazioni ad un gruppo ristretto. Ad esempio, 192.168.1.* indica che tutti gli indirizzi IP che iniziano per 192.168.1 sono autorizzati a fare il push. L'utilizzo del solo * indica niente restri-

ESEMPIO DI TIMER PROGRAMMATO

Nel CD allegato alla rivista trovate il codice di un'applicazione che utilizza un timer programmato. Questo programma è abbastanza semplice ma utilizza tutte le potenzialità relative ai timer programmati offerte dalla classe PushRegistry. In sostanza l'applicazione permette di registrare un timer programmato ad un certa ora. All'ora prefissata l'applicazione si sveglia e visualizza un form nel display del cellulare. Tale form ha come titolo Hello World! e come contenuto la stringa ioProgrammo. Attraverso l'applicazione è possibile sia registrare un timer sia disattivarlo. Da notare che nel metodo startApp, relativo alla MIDlet dell'applicazione, non è stato fatto alcun controllo per verificare se l'applicazione sia stata risvegliata dell'utente o dal timer programmato. Nei contesti in cui ciò è necessario potete utilizzare il metodo descritto alla fine dell'articolo. La classe che si occupa di registrare e disattivare il timer è *AlarmManager* e quello che segue è il codice dei suoi due metodi principali, ovvero i metodi per registrare e resettare un timer programmato. Il resto del codice (abbastanza commentato) lo trovate nel CD allegato alla rivista.

```
//registra il timer
private boolean registerAlarm(Date d)
{
    boolean ret = false;
    //recupera il  timestamp
    long t = d.getTime();
    //se la data (e ora) scelta non è
    //successiva a quella corrente
    //mostra un alert e restituisce false
    if(isValid(t) == false)
    {
        midlet.displayNotice(
            previousDisplay, "Il timer non
                deve essere antecedente
                    all'ora corrente");
        return ret;
    }
    //registra il timer
    try{
        String midletName =
            midlet.getClass().getName();
        PushRegistry.registerAlarm(
                        midletName, t);
        ret = true;
    }
    catch(Exception e)
    {
        System.err.println(e);
    }
    finally
    {
        return ret;
    }
}
//disattiva il timer
private void unregisterAlarm()
{
    try
    {
        String midletName =
            midlet.getClass().getName();
        PushRegistry.registerAlarm(
                        midletName, 0);
    }
    catch(Exception e)
    {
        System.err.println(e);
    }
}
```

zioni, ovvero chiunque può fare il push per risvegliare la MIDlet.

Alcuni esempi completi sono i seguenti:

```
MIDlet-Push-1: socket://:8000,
        j2meprojects.pushtechnology.PushMIDlet, *
MIDlet-Push-2: datagram://:7000, PushClass, *
```

Il primo registra una connessione di tipo socket sulla porta 8000. Il secondo una connessione di tipo datagramma sulla porta 7000. Entrambi permettono a chiunque di fare il push. C'è da dire che i timer programmati non supportano la registrazione statica ma solo quella dinamica, la quale risulta una scelta abbastanza ovvia.
La registrazione dinamica ci permette di comunicare, programmaticamente, la richiesta della MIDlet all'AMS. La registrazione dinamica avviene per mezzo di una classe definita in MIDP 2.0 il cui nome è PushRegistry. I metodi esposti da tale classe sono illustrati in **Figura 1**.

```
PushRegistry
+getFilter(String):String
+getMIDlet(String):String
+listConnections(boolean):String[]
+registerAlarm(String, long):long
+registerConnection(String, String, String):void
+unregisterConnection(String):boolean
```

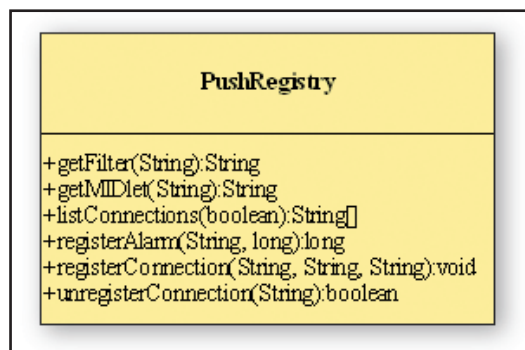



Fig. 1: ***La classe PushRegistry***

Il metodo getFilter restituisce una stringa rappresentante le restrizioni per questa registrazione. Il metodo getMIDlet restituisce il nome della MIDlet registrata. Gli altri metodi li vedremo nel prosieguo del presente articolo. Esempi di registrazione dinamica sono:

```
PushRegistry.registerConnection("datagram://:7000",
                this.getClass().getName(), "*");
PushRegistry.registerAlarm(this.getClass().getName()
                , (new Date()).getTime() + time);
```

La prima registra una connessione di tipo datagramma sulla porta 7000. La MIDlet risvegliata al verificarsi dell'evento sarà costituita dalla MIDlet corrente (specificata tramite this.getClass().getName()). La seconda registra un timer programmato che risveglierà la MIDlet corrente dopo time millisecondi.

CONNESSIONE DI TIPO SOCKET

Vediamo, ora, un esempio che illustra come registrare una connessione di tipo socket utilizzando il metodo registerConnection della classe PushRegistry, effettuando quindi una registrazione dinamica. Il codice da utilizzare per fare ciò è il seguente:

```
...
String currentMIDlet = this.getClass().getName();
String url = "socket://:8000";
String filter = "*";

try {
    ServerSocketConnection ssc =
        (ServerSocketConnection)Connector.open(url);
    PushRegistry.registerConnection(
                url, currentMIDlet, filter);
    SocketConnection sc =
            (SocketConnection)ssc.acceptAndOpen();
    InputStream is = sc.openInputStream();
    // Quì vengono letti e manipolati i dati
    // ricevuti dal server
}
catch(Exception e)
{
    System.out.println("Eccezione: " + e.toString());
}
...
```

Come si può vedere viene aperta una connessione di tipo socket sulla porta 8000 senza restrizioni su chi può effettuare il push. Chiamando il metodo registerConnection viene, poi, registrata la connessione. Così facendo, appena si esce dalla MIDlet in questione, l'AMS gestirà le connessioni di tipo

IL NOSTRO ESEMPIO

Per potere realizzare l'esempio proposto all'interno dell'articolo é necessario procurarsi il Wireless Toolkit 2.2. Si tratta del tool proposto da Sun per la realizzazione di applicativi per i cellulari. é possibile scaricarlo all'indirizzo http://java.sun.com/products/sjwtoolkit/download-2_2.html. Il Wireless Toolkit 2 implementa alcune caratteristiche piuttosto interessanti. Intanto supporta il profilo MIDP 2.0, poi le Wireless Messaging API 2.0 e le API per il bluetooth. Il toolkit dunque copre la quasi totalità delle applicazioni dedicate al mobile. All'interno del toolkit é presente un emulatore completo che vi svincola dalla necessità di provare l'applicazione su un telefono cellulare e vi mette in grado di effettuare una serie di test che altrimenti sarebbero impossibili vista la grande quantità di periferiche mobili attualmente disponibili e che supportano applicazioni Java.

socket che avvengono sulla porta 8000 attivando la MIDlet all'occorrenza. Cioè, l'AMS, anche quando la MIDlet non è attiva, rimarrà in ascolto di connessioni di rete di tipo socket e risveglierà la MIDlet, chiamando il suo metodo startApp non appena rileverà la connessione di cui sopra. Il metodo acceptAndOpen restituisce un oggetto di tipo StreamConnection (ecco il perché del cast a SocketConnection) che rappresenta una connessione socket lato server. È da notare che la chiamata è di tipo bloccante e quindi, in un'applicazione reale, conviene effettuarla in un thread separato. Appena stabilita la connessione viene aperto un stream di input per ricevere i dati dal server e poi vengono fatte le opportune manipolazioni sui dati così ricevuti. È da notare che le possibili eccezioni vengono gestite in modo generale con Exception. In un'applicazione per il mondo reale, ogni eccezione dovrebbe essere gestita in modo appropriato. In particolare, le eccezioni che possono essere lanciate sono di tre tipi: SecurityException, ClassNotFoundException e IOException. La prima viene lanciata se la MIDlet non ha i permessi sufficienti per effettuare connessioni di rete. La seconda quando, ad esempio, non appena si verifica la connessione l'AMS non trova la MIDlet specificata in registerConnection. L'IOException si può verificare quando, ad esempio, la porta specificata è già in uso.

CONNESSIONE DI TIPO DATAGRAMMA

La registrazione di una connessione di tipo datagramma viene fatta più o meno allo stesso modo di quella di tipo socket. Si può evincere ciò dal seguente pezzo di codice:

```
...
String currentMIDlet  = this.getClass().getName();
String url = "datagram://:9000";
String filter = "*";

try
{
    UPDDatagramConnection udc =
        (UDPDatagramConnection)Connector.open(url);
    PushRegistry.registerConnection(url,
                                    currentMIDlet, filter);
    Datagram dg = udc.newDatagram(
                            udc.getNominalLength());
    udc.receive(dg);
    while (!done)
    {
        udc.receive(dg);
        // Quì vengono letti e manipolati i dati
        // ricevuti dal server. Per ricevere i dati si
        // utilizza il metodo getData() della classe
        // Datagram e che restituisce un array di
        // byte rappresentante il contenuto del
                                              buffer
    }
}
catch(Exception e)
{
    System.out.println("Eccezione: " + e.toString());
}
...
```

Viene aperta una connessione di tipo datagramma sulla porta 9000. Analogamente a prima viene, poi, registrata la connessione tramite la chiamata al solito metodo registerConnection della classe PushRegistry.

TIPI DI CONNESSIONI

In una connessione di tipo datagramma non è garantito né l'arrivo dei pacchetti né l'ordine in cui essi arrivano. Per tale motivo, questo tipo di connessione è detta packet-based, ovvero basata sui pacchetti. Quella di tipo socket, invece, è detta stream-based. Uno stream, in questo caso, è un flusso continuo d'informazione ed in una connessione di tipo socket l'informazione arriva in modo affidabile ed ordinato. La "pacchettizzazione" dell'informazione, tuttavia, offre, in alcuni contesti, delle performance migliori rispetto ad una connessione di tipo socket.

Il passo successivo è creare un oggetto di tipo Datagram. Ciò viene fatto chiamando il metodo newDatagram su udc e passando come parametro un intero rappresentante la dimensione necessaria del buffer per il datagramma. Questo valore viene ricavato chiamando il metodo getNominalLength su udc. Tale metodo restituisce la lunghezza nominale del datagramma. Quest'ultima si riferisce alla dimensione del datagramma memorizzato nel buffer. Il metodo receive si mette in attesa di una connessione di rete e appena rilevata riceve il datagramma inviato. Anche questo metodo, come acceptAndOpen del caso precedente, è di tipo bloccante quindi, nel caso reale, conviene utilizzare un thread separato per questo tipo di operazioni. Per ricevere i dati dal server si chiamerebbe, poi, il metodo getData sull'oggetto di tipo Datagram, nel nostro caso dg. Ad esempio:

```
byte[] contenutoBuffer = dg.getData();
```

Per le eccezioni valgono le stesse considerazioni fatte per la connessione di tipo socket.

Sia nel caso socket che in quello datagramma, per rimuovere una registrazione dinamica basta invocare il metodo unregisterConnection della classe PushRegistry.
Un esempio è:

```
PushRegistry.unregisterConnection(
                              "datagram://:9000");
```

TIMER PROGRAMMATO

Un timer programmato può risultare molto utile in diverse situazioni.
Registrare un timer programmato è abbastanza semplice.
Un esempio di codice è il seguente:

```
...
String  currentMIDlet = this.getClass().getName();
Date date = new Date();
try
{
  PushRegistry.registerAlarm(currentMIDlet,
                              date.getTime() + time);
}
catch(Exception e)
{
   System.out.println("Eccezione: " + e.toString());
...
```

Questo codice registra la MIDlet corrente per essere lanciata dopo time millisecondi. Se la MIDlet corrente era già stata schedulata precedentemente, il metodo registerAlarm restituisce un long rappresentante l'orario in cui era stata registrata per il push.
Questo long indica il numero di millisecondi trascorsi dalla mezzanotte del 01/01/1970. Se, invece, è la prima volta che la MIDlet si registra per il push allora registerAlarm restituisce 0.
Per rimuovere un timer programmato basta passare 0 al metodo registerAlarm, ad esempio:

```
PushRegistry.registerAlarm(currentMIDlet, 0);
```

Rimuove l'allarme registrato per la MIDlet indicata dalla stringa currentMIDlet.

CHI HA ATTIVATO LA MIDLET?

Rimane un piccolo problema da risolvere. Abbiamo visto che una MIDlet, in MIDP 2.0, può essere attivata manualmente (dall'utente) o da un evento esterno (come una connessione di rete). In entrambi i casi, l'AMS invoca il metodo startApp della MIDlet in questione. Il problema è che, se siamo interessati a sapere se la MIDlet è stata attivata dall'utente o da un evento esterno, lo dobbiamo scoprire programmaticamente.
Fortunatamente, la classe PushRegistry mette a disposizione un metodo dedicato proprio a questo; esso è listConnections.
Un esempio è il seguente:

```
public void startApp()
{
  String connectionList[];
  connectionList =
PushRegistry.listConnections(true);
  if(connectionList == null || connectionList.length
                                               == 0)
  {
    //attivata dall'utente
   }
   else
   {
    //attivata da un'attività di rete
   }
}
```

Il parametro true indica di considerare solo le connessioni che stanno inviando dei dati. Passando false verrebbero restituite tutte le connessioni registrate per la MIDlet corrente. È da notare che la classe PushRegistry non fornisce un metodo per scoprire se la MIDlet sia stata attivata da un timer. Se tale informazione è fondamentale si può utilizzare un artificio. Si potrebbe, ad esempio, memorizzare tramite RMS il timestamp del timer programmato e confrontarlo con quello ricavato all'avvio dell'applicazione.

CONCLUSIONI

In questo articolo abbiamo visto come poter attivare una MIDlet tramite un evento esterno. Questa nuova funzionalità, offerta da J2ME con MIDP 2.0, ci consente di sviluppare applicazioni, per dispositivi mobili, molto interessanti.
Ad esempio, si potrebbe realizzare un'applicazione client/server di messaggeria e sviluppare lato client un programma che interroghi il server periodicamente per vedere se ci sono nuovi messaggi.
Gli strumenti, ora, li abbiamo...rimane a noi mettere le idee!

Alessandro Lacava

SUL WEB

Ottimo articolo sulla push technology con un esempio completo su WMA
http://www.corej2me.com/DeveloperResources/IBM/pushtechnology.pdf

Usare XPATH per trovare i dati

XSL è un linguaggio di trasformazione molto potente che ci consente di trasformare file XML in qualsiasi altro formato. Vediamo come selezionare solo i dati che ci interessano per trasformarli

Come abbiamo visto nella parte terza del corso se da una parte la sintassi XSL è abbastanza semplice, dall'altra essa demanda parte del lavoro ad un altro linguaggio : XPATH. È un linguaggio per la ricerca semplificata di informazioni in un documento XML. Per coloro che hanno familiarità con i database potremmo dire che XPath rappresenta per la famiglia XSLT quello che la clausola *WHERE* con i relativi operatori rappresenta per SQL. Un documento XML è definibile come un albero di nodi che ha come una radice il primo di essi.

```
<radice>
        <figlio>testo</figlio>
        <figlio attributo="valore"/>
</radice>
```

In questo semplicissimo documento XML vediamo appunto che sotto al primo nodo (la radice) sono inseriti altri due nodi di tipo *Element* chiamati *"figlio"* questi a loro volta hanno rispettivamente al loro interno un *TextNode* il primo (ovvero il valore *"testo"*) ed un *AttributeNode* con un relativo valore.
Gli oggetti che incontriamo leggendo il documento sono quindi:

```
-ElementNode (radice)
        -       ElementNode(figlio)
                        TextNode(testo)
        -       ElementNode(figlio)
                        AttributeNode(valore)
```

Da questo ne deduciamo che i nodi *Element* possono racchiudere al loro interno altri *Element*, *Attribute* e *Text* e che ogni nodo ha una relazione con gli altri.

RELAZIONI TRA I NODI

Le relazioni tra i nodi seguono uno schema ben preciso. In particolare le elencheremo secondo la seguente classificazione:

- *Parent*
- *Children*
- *Siblings*
- *Ancestor*
- *Descendants*

Parent: Un nodo *Element* avrà una relazione di tipo parent con il nodo in cui è contenuto. Tutti i nodi (tranne quello radice), hanno un genitore, ad esempio:

```
<mondo>
        <europa>
                <italia/>
        </europa>
</mondo>
```

L'elemento *Parent* del nodo *<italia>* è *<europa>* che a sua volta ha come *Parent* *<mondo>*.

Children: I nodi di tipo *Element* possono avere uno o più nodi figli. Nell'esempio precedente l'elemento *<italia>* è *Children* di *<europa>*.

Siblings: I nodi di tipo *Element* hanno una relazione *Sibling* con gli elementi di pari livello.
Nell'esempio seguente sono tra loro *Siblings* *<europa>* e *<asia>* così come *<italia>* e *<francia>:*

```
<mondo>
        <europa>
                <italia/>
                <francia/>
</europa>
        <asia>
                <india/>
</asia>
</mondo>
```

Ancestors: Sono *Ancestors* (ascendenti) di un ele-

mento tutti gli elementi che lo precedono, fino alla radice (compresa). Ad esempio in:

```
<mondo>
        <europa>
                <italia/>
        </europa>
</mondo>
```

sono *Ancestors* di *<italia>* gli elementi *<europa>* (che è anche *Parent*) e *<mondo>*.

Descendants: Sono *Descendants* (discendenti) di un elemento tutti gli elementi che esso racchiude. Ad esempio in

```
<mondo>
        <europa>
                <italia/>
                <francia/>
        </europa>
</mondo>
```

sono *Descendants* di *<mondo>* gli elementi *<europa>*, *<italia>* e *<francia>*.

LA SINTASSI XPATH

Abbiamo già detto che *XPATH* è il linguaggio che ci consente di effettuare delle ricerche all'interno di un documento XML. Per i nostri esempi, per capire come utilizzare questo linguaggio, utilizzeremo il seguente documento di partenza:

```
<libreria>
    <libro id="1">
        <titolo lingua="en">Learnig XML</titolo>
              <autore>John Smith</autore>
              <prezzo>10.00</prezzo>
    </libro>
    <libro id="2">
        <titolo lingua="it">I Promessi Sposi
                                  </titolo>
        <autore>Alessandro Manzoni</autore>
        <prezzo>15.00</prezzo>
    </libro>
</libreria>
```

Supponiamo di voler ottenere tutti gli autori di libri contenuti nel documento XML di partenza. Un modo per farlo è il seguente:

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl=
        "http://www.w3.org/1999/XSL/Transform">
        <xsl:template match="/">
        <xsl:for-each select="//libro/autore">
```

```
                <xsl:value-of select="."/>
                </xsl:for-each>
        </xsl:template>
</xsl:stylesheet>
```

La trasformazione restituisce la lista degli autori:

- *John Smith*
- *Alessandro Manzoni*

Notiamo che la sintassi XPath è stata applicata in tre occasioni:

1. Per definire l'ambito di applicazione del template con *match="/"* (*"/ "* rappresenta appunto la radice del documento).
2. Per definire gli elementi del ciclo *for-each* con *select="//libro/autore"* che sta a significare: prendi tutti i *Children* dell'elemento *<libro>* qualunque sia la sua posizione.
3. All'interno del ciclo, quando ormai l'elemento corrente è *<autore>* per stampare il valore di testo contenuto nel nodo corrente *(".")*.

Da notare che il riferimento al nodo corrente espresso con il punto *(".")* a seconda del contesto può rappresentare il valore del testo contenuto dal nodo, ma può essere usato anche per riferirsi al nodo corrente. Se all'interno del ciclo *for-each*, quando il contesto era il nodo *<autore>* avessimo scritto:

```
<xsl:value-of select="./../titolo/@lingua"/>
```

il punto *(".")* sarebbe stato il nodo corrente e i due punti *("..")* il *Parent* e quindi, partendo dal nodo corrente *(<autore>)* saremmo risaliti a *<libro>* e ridiscesi all'elemento *<titolo>* prendendo il suo attributo *"lingua"*.

PREDICATI

I predicati sono usati per trovare un nodo specifico o un nodo che contiene (o ha un attributo che contiene) un dato valore. I predicati devono essere contenuti in parentesi quadre [ ].
Esempi di predicati possono essere:

path expression	Risultato
/libreria/libro[1]	Seleziona il primo nodo <libro> sotto a <libreria>
/libreria/libro[last()]	Seleziona l'ultimo nodo <libro> sotto a <libreria>
//titolo[@lingua='en']	Seleziona i nodi <titolo> che hanno l'attributo "lingua" con valore "en"
//libro[titolo/@lingua='en']/autore	Seleziona i nodi <autore> sotto <libro> che ha l'attributo "lingua" con valore "en" nell'elemento <titolo>

PROVARE GLI ESEMPI

Potete creare il file XML di partenza e il corrispondente file XSL di trasformazione nella stessa directory e aggiungere le linea

```
<?xml version="1.0"?>
<?xml-stylesheet type=
        "text/xsl" href=
        "nomefile.xsl"?>
```

all'inizio del file XML. Oppure usando JavaScript:

```
<SCRIPT language =
        "javascript">
 function transform(){
    var srcTree = new
     ActiveXObject("Msxml2
      .DOMDocument.4.0");
    srcTree.async=false;
    srcTree.load(
        "helloworld.xml");
    var xsltTree= new
     ActiveXObject("Msxml2
     .DOMDOCUMENT.4.0");
    xsltTree.async =
false;
    xsltTree.load(
        "helloworld.xsl");

    return srcTree
        .transformNode(
            xsltTree);
 }
</SCRIPT>
```

APPLICAZIONE DEI CONCETTI IN XSL

Vogliamo estrarre tutti i titoli dei libri che costano meno di 11 euro:

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl=
            "http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<xsl:for-each select="//libro[prezzo &lt; 11]/titolo">
   <xsl:value-of select="."/>
   </xsl:for-each>
   </xsl:template>
</xsl:stylesheet>
```

Il risultato sarà: *Learnig XML*. Da notare che l'espressione *"prezzo < 11"* è stata scritta *"prezzo < 11"* utilizzando l'entità *<* per rappresentare il segno di minore *(<)*. Ciò perchè secondo la sintassi XML > e < non possono essere usati all'interno degli attributi (vedi parte terza del corso).

SELEZIONARE NODI NON CONOSCIUTI

Nel caso in cui si vogliano effettuare delle ricerche all'interno di nodi il cui nome non è conosciuto si possono usare dei metacaratteri *(wildcards)*, ad esempio:

Espressione	Descrizione
*	Qualsiasi elemento
@*	Qualsiasi attributo
node()	Qualsiasi nodo di ogni tipo

La sintassi */libreria/** seleziona tutti gli elementi sotto *<libreria>* indipendentemente dal nome.

SELEZIONI MULTIPLE

L'utilizzo dell'operatore "|" in Xpath consente selezioni in più percorsi. Corrispone al classico *OR* logico. Ad esempio:

path expression	Risultato
//libro/titolo | //libro/autore	Seleziona sia <titolo> che <autore> sotto a <libro>
/libreria/libro /titolo | //prezzo	Seleziona sia <titolo> sotto a <libro> che tutti i nodi <prezzo>

GLI OPERATORI XPATH

Fin qui abbiamo quasi sempre usato un operatore senza averlo detto esplicitamente. In particolare abbiamo utilizzato l'operatore "//" che restituisce un insieme di nodi. Nel nostro caso lo abbioamo usato all'interno di una *select*. L'operartore "//" non è l'unico disponibile in XPATH. In particolare è possibile costruire espressioni XPath costituite da operandi e operatori e che possono restituire valori di vario genere:

- **Node-set** - ovvero un insieme di nodi
- **String** - testo
- **Boolean** - vero o falso
- **Number** - numero

In **Tabella 1** la lista completa degli operatori.
Quanto si utilizzano gli operatori occorre ricordarsi che XPath ha una sintassi case-sensitive, cioè che fa differenza tra maiuscole e minuscole. Per cui l'espressione : *"prezzo > 9.00 AND prezzo < 9.90"* non funzionerebbe in quanto *"AND"* è scritto in maiuscolo mentre l'operatore è *"and"*, minuscolo.

Operatore	In espressioni XSL	Descrizione	Esempio	Valore di ritorno
|	|	Trova due o più node-set	//libro | //cd	Restituisce node-set con tutti i nodi <libro> e <cd>
+	+	Somma	6 + 4	10
-	-	Sottrazione	6 - 4	2
*	*	Moltiplicazione	6 * 4	24
div	div	Divisione	8 div 4	2
=	=	Uguale	prezzo=9.80	vero se il prezzo è 9,80 falso se il prezzo è 9,90
!=	!=	Non uguale	prezzo!=9.80	vero se il prezzo è 9,90 falso se il prezzo è 9,80
<	<	Minore di	prezzo < 9.80	vero se il prezzo è 9,00 falso se il prezzo è 9,80
<=	<=	Minore o uguale a	prezzo <=9.80	vero se il prezzo è 9,00 falso se il prezzo è 9,90
>	>	Maggiore di	prezzo > 9.80	vero se il prezzo è 9,90 falso se il prezzo è 9,80
>=	>=	Maggiore o uguale a	prezzo >= 9.80	vero se il prezzo è 9,90 falso se il prezzo è 9,70
or	or	OR logico	prezzo=9.80 or prezzo=9.70	vero se il prezzo è 9,80 falso se il prezzo è 9,50
and	and	AND logico	prezzo > 9.00 and prezzo < 9,80	vero se il prezzo è 9,80 falso se il prezzo è 8,50
mod	mod	Modulo (resto di divisione)	5 mod 2	1

Tabella 1: La lista completa degli operatori

FUNZIONI XPATH

Oltre agli operatori, in XPath è possibile utilizzare un ricco set di funzioni per incrementare la flessibilità e la potenza nell'estrazione dei dati.
Le funzioni si dividono per categorie:

- **Node-Set** - Accettano un *node-set* come argomento, ritornano un *node-set*, o restituiscono in-

formazioni su un particolare nodo all'interno di un *node-set*.

- **String** - Compiono valutazioni, formattazioni, e manipolazioni su stringhe.
- **Boolean** - Valutano l'espressione argomento per ottenere un risultato Booleano (vero o falso).
- **Number** - Valutano l'espressione argomento per ottenere un risultato numerico.

Tutte le funzioni hanno la seguente sintassi:

```
nomefunzione([espressione])
```

L'espressione da passare come argomento può essere anche assente, in questo caso comunque andranno sempre utilizzate le parentesi vuote *"()"*.

FUNZIONI NODE-SET

Di seguito analizziamo una per una le funzioni che appartengono a questa categoria:

- **count:** Restituisce il conteggio dei nodi individuati con l'espressione.

```
<xsl:if test="count(//libro) &gt; 3">
...
</xsl:if>
```

restituisce il numero di nodi *"Libro"*

- **position:** Restituisce la posizione del nodo corrente nella relativa lista di nodi.

```
<xsl:for-each select="//libro">
  <div>
      <xsl:value-of select="position()"/>.
                  <xsl:value-of select="titolo"/>
  </div>
</xsl:for-each>
```

Che dà come risultato:

1. primo titolo
2. secondo titolo ecc.

- **last:** Restituisce il numero corrispondente all'ultima posizione in una lista di nodi. Da utilizzare nel contesto di un predicato per estrarre l'ultimo nodo di una lista.

```
<xsl:for-each select="//libro[last()]">
    <xsl:value-of select="titolo"/>
</xsl:for-each>
```

Oppure

```
<xsl:for-each select="//libro">
  <xsl:if test="position() = last()">
    <xsl:value-of select="titolo"/>
  </xsl:if>
</xsl:for-each>
```

- **name:** Restituisce il nome del nodo passato come argomento quando è usata nella forma:

```
name(node-set)
```

oppure restituisce il nome del nodo corrente quando è usata nella forma:

```
  name()
<xsl:for-each select="/*">
  <!-- stampa il nome del nodo radice -->
    <xsl:value-of select="name()"/>
</xsl:for-each>
```

Oppure direttamente

```
<xsl:value-of select="name(/*)"/>
```

Altre funzioni *node-set* meno usate sono:

- **id** - Seleziona gli elementi per il valore dell'attributo *"id"* (minuscolo). Ma non funziona allo stesso modo con tutti i motori per cui è preferibile usare l'espressione equivalente: *//*[@id='valore']*.
- **local-name** - che restituisce il nome locale di un nodo con un prefisso di namespace. Ad esempio nel caso di un nodo *<xsl:template>* *local-name* restituirà *"template"*. L'utilizzo è identico a *name*.
- **namespace-uri** - che restituisce il namespace URI di un nodo per cui viene impostato un namespace. Ad esempio nel caso di un nodo *<xsl:template>* *namespace-uri* restituirà *"http://www.w3.org/1999/XSL/Transform"*.

FUNZIONI STRING

Le funzioni di tipo string vengono utilizzate per manipolare il testo. Alcune delle più utilizzate sono

- **concat:** concatena due o più stringhe passate come argomenti *concat('abc', 'd', 'ef', 'g') abcdefg*
- **contains:** determina se una stringa è contenuta in un'altra.

Se come argomenti vengono passati riferimenti a nodi di essi viene preso il valore e interpretato come stringa

- **normalize-space**: elimina gli spazi bianchi in eccesso da una stringa o dal valore di un nodo

Utilizza questo spazio per le tue annotazioni

interpretato come stringa *normalize-space* ('una stringa') una stringa

- **starts-with** Restituisce vero o falso se una stringa inizia con il valore passato come secondo argomento.
- **string** converte un oggetto in una stringa
- **string-length** Restituisce la lunghezza in caratteri di una stringa
- **substring** Restituisce la sottostringa all'interno del primo argomento iniziando dalla posizione indicata nel secondo argomento (partendo da 1) per una lunghezza uguale a quella indicata nel terzo argomento
- **substring-after** Restituisce la sottostringa all'interno del primo argomento successiva alla stringa indicata nel secondo argomento
- **substring-before** Restituisce la sottostringa all'interno del primo argomento precedente alla stringa indicata nel secondo argomento
- **translate** Restituisce la sottostringa all'interno del primo argomento sostituendo i caratteri del secondo argomento con quelli del terzo.

FUNZIONI BOOLEAN

Si tratta delle funzioni classiche utilizzate per ottenere valori di tipo *true* o *false*

- **boolean** converte un espressione in un valore booleano (vero o falso) secondo questi criteri:

  * se l'argomento è un numero restituisce *false* se è 0 altrimenti *true*.
  * se l'argomento è una stringa restituisce *false* se è vuota altrimenti true.
  * se l'argomento è un riferimento a un node-set restituisce *false* se il node-set è vuoto altrimenti *true*.

- **not** - Restituisce *true* se l'argomento è *false* e viceversa. Ad esempio *"not(1 > 2)"* dà false perché l'espressione *1>2* è *true*.

Altre funzioni Boolean sono:

- **false()** e **true()** - che restituiscono sempre *false* e *true*
- **lang(nome)** - che restituisce true o false a seconda se l'attributo *xml:lang* del nodo corrente corrisponda o meno a quello passato come argomento.

FUNZIONI NUMBER

La sintassi XPath supporta le funzioni su numeri che possono essere usate con gli operatori di confronto nelle espressioni filtro.

- **ceiling** - Arrotonda un numero decimale all'intero superiore più prossimo.
  Ad esempio: *ceiling(5.5)* restituisce 6
- **floor** - Arrotonda un numero decimale all'intero inferiore più prossimo.
  Ad esempio: *floor(5.5)* restituisce 5
- **round** - Arrotonda un numero decimale all'intero superiore o inferiore più prossimo.
  Ad esempio:

  round(5.6) restituisce 6
  round (5.5) restituisce 6
  round (5.4) restituisce 5

- **number** - Converte l'argomento in numero, se l'argomento non può essere convertito restituisce *NaN (not a number)*
  Ad esempio:

  number('2048') restituisce 2048
  number('text') restituisce *NaN*

- **sum** - Utilissima funzione che accetta un node-set come argomento, converte i valori dei nodi (se può) in numeri e restituisce la somma.

Se abbiamo, ad esempio un file XML come quello che abbiamo utilizzato in precedenza

```
<libreria>
  <libro id="1">
    <titolo lingua="en">Learnig XML</titolo>
    <autore>John Smith</autore>
    <prezzo>10.00</prezzo>
  </libro>
<libro id="2">
  <titolo lingua="it">I Promessi Sposi</titolo>
    <autore>Alessandro Manzoni</autore>
      <prezzo>15.00</prezzo>
  </libro>
</libreria>
```

L'espressione sum(*//prezzo*) restituirà 25.

CONCLUSIONI

Questa puntata del corso è stata forse un po' più tecnica del solito. C'è da dire però che una buona conoscenza di XPath è fondamentale per utilizzare XSL perché le tecniche di estrazione dei dati sono il vero punto di forza di questa tecnologia. Nella prossima puntata parleremo invece di come XSL si integra con le tecnologie "ospiti" ed in particolare vedremo alcuni esempi concreti di XPath in azione.

Francesco Smelzo

SOFTWARE SUL CD

PHPMYADMIN 2.7.0

L'INTERFACCIA DI GESTIONE DI MYSQL

In molti hanno provato a rimpiazzare PHPMyAdmin come utility per la gestione di MySQL, ma fino ad ora PHPMyAdmin continua ad essere il più funzionale e il più utilizzato. La sua forza sta prima di tutto nell'essere una web application, e questo ha fatto si che diventasse un tool fornito di default dalla maggior parte degli hoster, in secondo luogo si tratta di

un'applicazione realmente ben fatta, la migliore per quantità e qualità delle funzioni esportate, anche quando messa a confronto con le applicazioni Desktop di pari fascia che in teoria dovrebbero essere avvantaggiate dalla maggiore flessibilità offerta dal Desktop rispetto al Web.
La versione 2.7.0 rappresenta un ennesimo passo importante e supporta già tutte le funzionalità del neonato MySQL 5.0

Directory:/ PHPMyAdmin

B2EVOLUTION 1.6

UNO DEI SISTEMI DI BLOGGING PIÙ COMPLETO

Grafica accattivante e modularità, queste sono le due caratteristiche essenziali di b2Evelotion, una piattaforma di blogging completa quanto efficace.

Completa perché il set di funzioni esposte è veramente elevato, si va

Jbuilder 2005 Foundation Edition

UNO STRAORDINARIO IDE RAD PER JAVA

Borland è famosa per essere stata la prima Software House a immettere sul mercato un ambiente di programmazione RAD. Poter disegnare l'interfaccia grafica e programmare ad eventi è stato uno dei passi fondamentali per la creazione di un nuovo modo di concepire la programmazione. A questa logica non poteva sfuggire Java, ma è significativo che uno dei pochi tool Rad per la programmazione Java venga proprio da Borland. Jbuilder consente di disegnare l'interfaccia con un comodo Visual Editor e poi di programmare gli eventi associati ai singoli elementi. Certo Java non si presta facilmente a questo genere di interpretazione, per cui l'ambiente di disegno è lontano dalle sofisticate caratteristiche a cui ci hanno abituato Delphi o C++ Builder ma rappresenta comunque un aiuto fuori dal comune per la programmazione di un'applicazione completa.

Il suo essere RAD non è comunque l'unico aspetto che contraddistingue l'ambiente, si tratta pur sempre di un editor di codice piuttosto sofisticato, dotato di sintax highlighting e code complexion, ma anche di altre funzioni. Ad esempio provando ad utilizzare un metodo non ancora definito l'ambiente in automatico ne scriverà la dichiarazione. Se provate a usare un metodo contenuto in un import automaticamente la libreria verrà importata anche se dimenticate di farlo a mano. Inoltre l'ambiente è molto leggero, sicuramente molto di più di quello di due diretti concorrenti: NetBeans ed Eclipse. La versione Foundation che qui presentiamo è liberamente utilizzabile.

Directory: /jbuilder

dall'inserimento delle informazioni direttamente in formato Html ad un granulare quanto sofisticato controllo dei permessi. b2Evolution si presta inoltre a essere multiblog, pochi clic di mouse vi metteranno in grado di crere un aggregatore efficace. Al solito la piattaforma scelta è quella di mySQL + PHP e questo fa si che il numero di hoster preparati ad ospitare questo tipo di piattaforma sia elevato. I costi sono nulli, essendo un software OpenSource

Directory: /b2evolution

USEBB 0.7

UN OTTIMO FORUM

Una Web Application che implementa un completo forum a disposizione degli utenti. Implementa tutte le caratteristiche tipiche di applicazioni di questo genere, dalla gestione dei permessi alla moderazione dei singoli argomenti, alla capacità di ricerca. Si tratta di una release abbastanza giovane ma che per quantità di funzioni può rivaleggiare con i più noti forum di questo genere.

Directory:/ UseBB

JOOMLA 1.0.4

L'EREDE DI MAMBO

Per lungo tempo Mambo è stato uno dei CMS più scaricati dalla rete. I motivi del successo di questo CMS erano da ricercarsi nella facilità di installazione dei moduli come nella semplicità di sviluppo di nuovi temi, molto personalizzabili.

Da una costola di Mambo e precisamente da un gruppo di sviluppatori in disaccordo con le politiche portate avanti da Miro la società che inizialmente ha sviluppato Mambo è nato Joomla.

Questo nuovo progetto ha una connotazione fortemente OpenSource che lo distingue da Mambo ed è stato accolto con entusiasmo dalla community internazionale, presentando le stesse identiche credenziali del suo illustre predecessore.

Directory:/ Joomla

DRUPAL 4.7.0

UNO STANDARD ORMAI PER QUANTO RIGUARDA I BLOG

Themeable, modulare, ricco di funzioni integrate. Queste le caratteristiche essenziali di Drupal, che non a caso nel corso del tempo ha conquistato una vasta platea di utilizzatori. Dal punto di vista del programmatore questo sistema di blogging si presenta particolarmente interessante perché espone una serie di API che consentono di creare i propri moduli con rapidità ed efficacia.

La community degli utenti/sviluppatori Drupal, inoltre è abbastanza vasta da consentire di godere di grande supporto e collaborazione. Una ghiotta opportunità dunque per chi lavora nel campo del Content Management System. Il linguaggio di riferimento è sempre PHP mentre c'è un'astrazione abbastanza elevata dai database.

Directory: /Drupal

WORDPRESS 1.5.2

IL PIÙ LEGGERO DEI BLOG

WordPress completa la schiera dei software di Blog Application presenti in questo numero di ioProgrammo. Si tratta probabilmente della più "antica" applicazione di blogging comparsa sul mercato. Il suo punto di forza è probabilmente la leggerezza, il suo punto di debolezza è probabilmente un'eccessiva semplificazione dell'interfaccia che talvolta rende difficile il reperimento di alcune funzionalità interessanti.

Directory:/ Wordpress

OSCOMMERCE 2.2

UN SISTEMA DI COMMERCIO ELETTRONICO COMPLETO

OsCommerce è stata una delle prime Web Application ad occuparsi di Ecommerce. Si tratta di un software scritto in PHP con l'ausilio di MySQL. Offre una completa interfaccia all'utente che viene portato dalla vetrina all'acquisto con un pipeline semplice quanto efficace.

Sono state implementate tutte le funzionalità di un'applicazione tipica di commercio elettronico, dal carrello personalizzato al pagamento per mezzo di gateway verso i circuiti più noti, inoltre sono presenti diverse funzionalità avanzate quali la gestione degli sconti e del magazzino, per non parlare di qualche funzione di cross selling. Si tratta dunque di un'applicazione molto ben congegnata. Unico suo limite, il fatto che lo sviluppo è fermo alla versione 2.2 da molto tempo, tuttavia i moduli aggiuntivi continuano a essere sviluppati da una folta community di sviluppatori, segno che il progetto è ancora vivo e utilizzato.

Directory: /OsCommerce

WXDEV CPP

UN DEV C++ POTENZIATO

Un'installazione semplificata che vi consente di usare subito Dev C++ con le estensioni dedicate alle WxWidgets, tali che lo rendono un ambiente rad e pronto per essere usato per la produzione di applicazioni multipiattafor-

ma. Ovviamente l'installazione proposta è relativa all'ambiente Windows ma le WxWidgets sono disponibili anche in ambiente Linux e tutto ciò rende questo tipo di configurazione molto appetibile anche per la creazione di applicazioni cross platform

Directory:/ wxWidgets

MYSQL 5.0.16

LA NUOVA VERSIONE DEL DB PIÙ USATO IN RETE

Stored Procedure, Trigger, queste le novità più interessanti di MySQL nella sua versione 5.0 arrivata sul mercato da qualche mese. La nuova versione è stata salutata con entusiasmo dal pubblico degli utilizzatori. Di fatto la migrazione dalle versioni precedenti non comporta difficoltà eccezionali a fronte dell'implementazione di una serie di novità che invece rivoluzionano il modo di utilizzare MySQL e ne velocizzano anche di molto l'esecuzione delle query di inserimento/selezione dati.

Si tratta di una svolta significativa per il database più usato in rete, che adesso a ragione si pone come un grande antagonista dei suoi rivali più blasonati.

Directory: /Mysql

J2SE 1.5.0

TUTTO IL NECESSARIO PER PROGRAMMARE IN JAVA

J2SE contiene il completo SDK che vi mette in grado di programmare in Java, dalle librerie, al compilatore, agli strumenti indispensabili. Si tratta di uno strumento maturo, ormai testato e utilizzato largamente. Basti dire che proprio di recente il numero di progetti Java su SourceForge ha superato quello relativo a C++. Si tratta ovviamente dello strumento indispensabile per poter programmare in Java

Directory: /J2SE

PHP 5.1.1

LA NUOVA VERSIONE DEL LINGUAGGIO PRINCIPE PER IL WEB

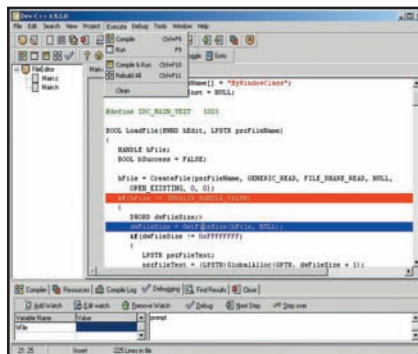
La versione 5.0 aveva segnato un passo di discontinuità eccezionale rispetto alla precedenti della serie 4.X, con l'introduzione di un modello ad oggetti complesso quanto insolito per PHP.

La versione 5.1 corregge una serie di Bug e modifica il comportamento di alcuni costrutti come ad esempio quelli relativi a "for each", e rappresenta un nuovo passo verso la futura release 6.0 di cui già circolano le prime indiscrezioni

Directory:/ PHP

DEV C++ 4.9.9.2

L'AMBIENTE EVOLUTO PER I PROGRAMMATORI C++

Dev C++ ormai lo conosciamo tutti, da lungo tempo viene ospitato nel cdrom allegato ad ioProgrammo. Si tratta di un IDE leggero quanto completo dedicato ai programmatori C++.

Dotatto di Sintax Higlighting, code Completion e una serie di features che aumentano di molto la produttività dei programmatori C++, viene distribuito in abbinata con il compilatore MingGW, ma può essere utilizzato efficacemente con la quasi totalità dei compilatori C++ oggi disponibili.

Directory: /devCpp

POSTGRESQL 8.1

IL PIÙ PROFESSIONALE

Di PostgreSQL avevamo avuto modo di occuparci su ioProgrammo nei numeri precedenti con un bell'articolo approfondito. PostgreSQL è realmente un DB che dovrebbe occupare una posizione di maggiore rilievo nella scala dei database utilizzati a livello professionale. Non ha davvero nulla da invidiare a DataBase dal calibro di Oracle o di MSSql server e per velocità e numero di funzioni esposta si pone ad un livello molto più alto anche rispetto a MySQL 5.0 che comunque ha colmato parte del gap che aveva con l'ultima release.

A fare la forza di MySQL è la completezza delle funzioni esposte che vanno dalle Stored Procedure, ai trigger, alla programmazione per mezzo di funzioni e alla modularità. Un grande database al quale manca forse soltanto una maggiore spinta di marketing per essere il leader del mercato

Directory:/ PostgreSQL

ECLIPSE 3.1.1

L'IDE TUTTOFARE

Eclipse è l'ambiente "configurabile" che si presta come IDE evoluto verso la quasi totalità dei linguaggi.

Ha trovato la sua collocazione ideale come IDE per Java ma lentamente sta diventando una sorta di raccoglitore del lavoro giornaliero di una moltitudine di programmatori, che proprio per la sua capacità di assolvere da un'unica interfaccia a più di un compito, lo utilizzano come unico riferimento nel corso delle proprie attività di sviluppo. Nel progetto Eclipse inoltre stanno confluendo nel tempo nomi importanti quali ad esempio Ibm e Borland, e questo garantisce uno svi-

luppo incessante e professionale di questo interessante strumento
Directory:/ Eclipse

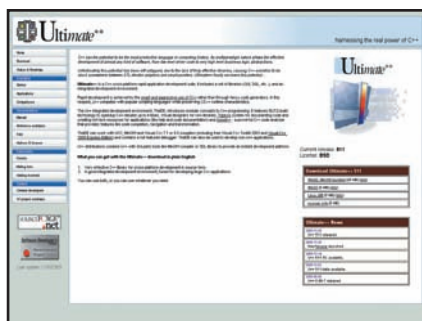
THING 0.1

L'EDITOR DI THINLET

Thinlet è una libreria Java che consente di creare interfacce grafiche utilizzando XML. Le interfacce così create vengono parserizzate dalla libreria e producono in output l'applicazione corretta. Thing è un editor XML per lo sviluppo rapido e visuale di interfacce compatibili con Thinlet. Si tratta di un editor scritto in Java, perciò perfettamente funzionante sia con Linux che con Windows
Directory:/ Thing

ULTIMATE ++ 5.1.1

L'IDE PIÙ INNOVATIVO PER C++

Se DEV C++ rappresenta ormai uno standard per i programmatori C++, Ultimate++ si sta dimostrando un'ottima alternativa.

Leggero, veloce, dotato di alcune estensioni che lo rendono in parte RAD, viene distribuito con il compilatore MingGW. È dotato naturalmente di tutte le caratteristiche di un buon IDE, tuttavia dispone di un'organizzazione che lascia rapidamente intendere che da questo ambiente dovremmo attenderci elle grosse novità per il futuro, sia in termini di prestazioni che di completezza.
Directory:/ Ultimatepp

OPEN REPORT 1.1.1

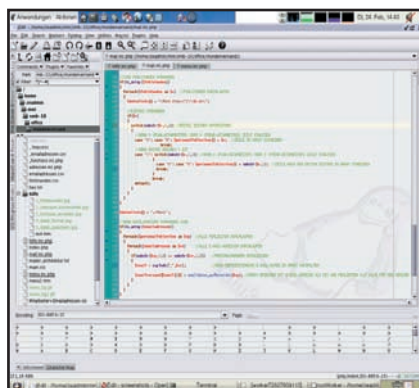
REPORT PER TUTTI I GUSTI

OpenReport è un sistema di reportistica completo e OpenSource. Il prodotto si presenta come molto efficace e dotato di tutte le funzioni tipiche di un tool per il report dei dati. Uno dei

suoi maggiori punti di forza consiste nel poter salvare la definizione del report direttamente in formato XML e questo lo rende particolarmente adatto ad interfacciarsi con un gran numero di applicazioni. D'altra parte il suo orientamento è quello di lavorare in congiunzione con PostgreSQL anche se non dovrebbe essere complesso modificarlo per renderlo compatibile con altri tipi di db. Il suo codice è opensource e modulare, quindi se trovate che i suoi meccanismi interni siano di vostro gradimento potete sempre modificarlo per aggiungere le parti che possono aiutarvi nel vostro lavoro
Directory:/ OpenRpt

JEDIT 4.2

IL PIÙ LEGGERO DEGLI EDITOR JAVA

In poco meno di un mega, un concentrato essenziale di funzioni che facilitano la programmazione di applicazioni Java.

Si tratta di un editor veramente leggero, ottimale per coloro che non hanno bisogno di grandi ambienti o di funziono avanzate ma più semplicemente di un aiuto rapido nel corso della stesura del codice
Directory: /Jedit

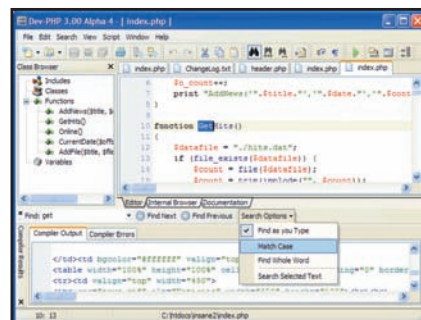
ITEXT 1.3.6

IL LEADER PER LA CREAZIONE DI PDF IN JAVA

iText è un grande strumento che da tempo è diventato il punto di riferimento centrale per i programmatori Java che devono produrre PDF.
Leggera, semplice, ben strutturata, la libreria non solo garantisce un utilizzo immediato senza necessità di grandi curve di apprendimento, ma riesce a produrre PDF estremamente piccoli, e tali da poter essere utilizzato con efficacia in un gran numero di circostanze.
Directory: /itext

DEV PHP 2.012

Un ide completo per i PROGRAMMATORI PHP

Si ok, per scrivere codice PHP è sufficiente il notepad, ma certamente un buon IDE aiuta.

DevPHP è nato senza troppo rumore già da molto tempo, ma nel corso degli anni è andato affermandosi come uno dei migliori IDE in circolazione per i programmatori PHP. Oltre alle normali caratteristiche di cui devono essere dotati gli ambienti di programmazione, spicca anche l'integrazione di un debugger e di un web server che aiuta e non di poco nella programmazione delle applicazioni
Directory: /DevPHP

PYTHON 2.4.2

IL LINGUAGGIO EMERGENTE

Python è in America il linguaggio che più di ogni altro nel corso del 2005 ha incrementato il numero dei suo utilizzatori. Si tratta di un linguaggio elegante, ad oggetti, completo nelle sue funzionalità e multipiattaforma.
Queste le caratteristiche che fanno di Python un linguaggio decisamente

comodo da utilizzare in una smisurata quantità di applicazioni. Il suo unico neo risiede probabilmente in una curva di apprendimento non brevissima, tuttavia le difficoltà iniziali sono compensate dall'aumento della produttività raggiungibile quando la piena conoscenza del linguaggio viene raggiunta. Proprio greazie alle sue caratteristiche, Python è diventato uno dei linguaggi più utilizzati per la per il controllo del sistema in ambiente Linux. Recentemente grazie al fatto di essere multipiattaforma sta trovando applicazioni concrete anche in ambiente Windows sia per la realizzazione di piccoli script che per l'automazione di compiti ripetitivi.

Directory:/ Python

IRLLICHT 0.14

L'ENGINE PER LA PROGRAMMAZIONE SEMPLIFICATA DI VIDEOGIOCHI

Da quando ioProgrammo ha iniziato a presentare Irrlicht con una bella serie di articoli proposta da Alfredo Marroccelli, a cui completamento è stato pubblicato il libro "Programmare Videogiochi" allegato a ioProgrammo 97 e dello stesso autore, Irrlicht è diventata un punto di riferimento

certo per un vasto numero di programmatori di VideoGames.

Il merito, certo, non è tutto di ioProgrammo, che anzi ha scelto questo engine per lo sviluppo di videogame proprio per le sue grandi caratteristiche, per la velocità e la semplicità delle funzioni esposte. Oggi grazie ad Irllicht sono nati un numero elevatissimo di nuovi videogame. Suppota C++ ma anche .NET, da provare...

Directory:/ irrlicht

AVI CREATOR 1.5

PER CREARE PICCOLE ANIMAZIONI DA INSERIRE NEI TUOI PROGRAMMI

Avi Creator appartiene a quella schiera di utility che non si possono definire esattamente dedicate ai programmatori ma che aiutano senza dubbio a creare interfacce esteticamente accattivanti e che restituiscono un maggiore feedback allutente. Si tratta di un software che riesce a creare delle minianimazioni in formato AVI assemblandole da una lista di bitmap. Utili per sottolineare ad esempio operazioni come la copia dei file.

Directory: /avicreator

FREEBASIC 15B

UN BASIC CON LICENZA GNU

Questo compilatore farà la felicità di un gran numero di affezionati, che si troveranno a disposizione un completo compilatore Basic a linea di comando esattamente come ai bei tempi dellinizio dellinformatica. Al di la dellaspetto nostalgico bisogna però sottolineare che FreeBasic risulta utilissimo nel confezionare piccole ma potenti utility in modo rapido che possono automatizzare il comportamento del sistema. Il compilatore viene distribuito con una serie impressionante di librerie che vanno dalle GD per la creazione di interfacce grafiche alle fastcgi per la creazione di piccoli script adatti al web

Directory: /Frebasic

JAMELEON 3.0.4

UN TOOL PER IL TEST AUTOMATICO DELLE APPLICAZIONI

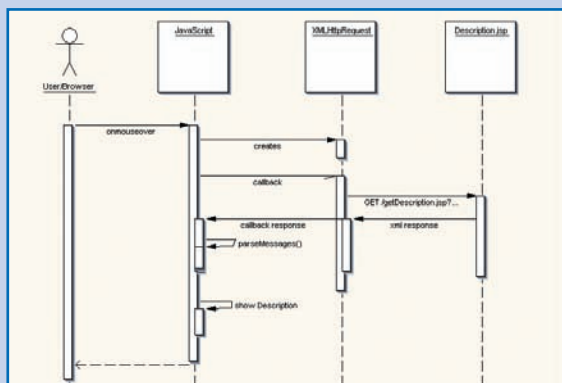
Jameleon è unapplicazione Intelligente per effettuare sessioni di test su applicazioni in fase di preproduzione. Si basa sul concetto di Keyword. Ad ogni Keyword corrisponde una Vista che rappresenta uno stato dellapplicazione. Modificando alcuni file java si possono poi simulare i diversi comportamenti dellutente e ottenere la risposta dellapplicazione. Tutto questo automatizza di molto le sessioni di testing

Directory: /Jameleon

MagicAjax.NET

PER INSERIRE FUNZIONALITÀ AJAX NELLE TUE APPLICAZIONI WEB

Ajax è una tecnologia sufficientemente recente, ma che sta cambiando radicalmente il modo di pensare alle applicazioni web. Sostanzialmente consente di non dover ricaricare una pagina intera quando un dato al suo interno cambia, ma solo la parte che interessa. Ad esempio volendo ottenere il numero di elementi presenti in un database, non è necessario effettuare lintero refresh di una pagina tramite una sessione di post, ma semplicemente in modo molto dinamico e sfruttando larchitettura interna del DOM si può ottenere il dato modificando a runtime letichetta che lo mostra. MagicAjax consente di inserire questo genere di funzionalità allinterno di applicazioni web .NET

Directory: /MagicAjax

Questione Sfida di menti sul Pi Greco

Da secoli gli scienziati si battono per trovare il maggior numero di cifre decimali possibili per il numero π. Vediamo quali sono gli algoritmi utilizzati dai moderni calcolatori

Il numero3,14159 26535 89793 23846 26433 83279 50288 41971 69399 37510 58209 74944 592 è soltanto un'approssimazione di π. Si tratta di un numero reale irrazionale, costituito da un numero infinito di cifre dopo la virgola. In geometria piana pi greco è definito come il rapporto tra la circonferenza e il diametro di un cerchio. In sintesi è la costante fondamentale che permette di calcolare circonferenza e area di un cerchio:

$$P_c = 2\pi r$$
$$A_c = 2\pi r^2$$

Ma lo scalare è fondamentale anche nella trigonometria. Rispetto ad essa pi greco è il più piccolo numero per il quale *sin(x)=0*. Ovviamente, pi greco è legato a moltissime formule sia della geometria sia dell'analisi. Figure come ellisse, cilindro, sfera e cono, e argomenti come la trigonometria, le serie e gli integrali, usano π. Risultano di particolare interesse alcune serie che sono state scoperte e che descrivono pi greco o frazioni di esso. La più antica è dovuta a Leibniz, che sarà oggetto di analisi più approfondite nei prossimi paragrafi. Per essa vale la relazione:

$$\frac{1}{2} - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \ldots = \frac{\pi}{4}$$

La particolarità di questa forma è l'alternanza di segno dei singoli termini. Un'altro interessante metodo per il calcolo di pi greco è:

$$\frac{2}{1} \cdot \frac{2}{3} \cdot \frac{4}{3} \cdot \frac{4}{5} \cdot \frac{6}{5} \cdot \frac{6}{7} \cdot \frac{8}{7} \cdot \frac{8}{9} \ldots = \frac{\pi}{2}$$

Dovuta a Wallis. In questo caso si tratta di un prodotto tra fattori. Ancora si ha la forma di Eulero che è la risposta al problema di Basel. In tal caso si ritorna ad una serie (somma tra termini).

$$\varsigma(2) = \frac{1}{1^2} \cdot \frac{1}{2^2} \cdot \frac{1}{3^2} \cdot \frac{1}{4^2} + \ldots = \frac{\pi^2}{6}$$

Tale costruzione è più efficace rispetto a quella di Leibniz proprio perché converge più rapidamente al valore della costante, ciò è dovuto alla presenza di sole somme. Un'espressione analoga è la conosciuta zeta di Riemann.

$$\varsigma(4) = \frac{1}{1^4} \cdot \frac{1}{2^4} \cdot \frac{1}{3^4} \cdot \frac{1}{4^4} + \ldots = \frac{\pi^4}{90}$$

Per terminare i sintetici riferimenti alla teoria riporto l'identità di Eulero, per molti considerata la più bella formula presente in matematica. Concetto, questo ribadito dal nobel Richard Feynman.

$$e^{\pi i} + 1 = 0$$

STORIA DI Π

Tale numero che viene naturalmente associato a figure circolari ha una storia molto ricca. Per ragioni pratiche in passato, fin dai tempi dei babilonesi era necessario calcolare aree circolari. A quei tempi il numero era approssimato a 3. Successivamente, tale valore è stato man mano "aggiustato", ossia è stato perfezionato fino ad arrivare al numero preciso che oggi conosciamo. Un numero dell'ordine di alcuni miliardi di cifre decimali, che giorno dopo giorno, potenti calcolatori associati a sofisticati software, tendono a farlo aumentare. Ma ritorniamo a qualche secolo fa. Si ha traccia di pi greco sia tra i babilonesi sia tra i cinesi. Uno studio più organico è dovuto al conosciuto Archimede che approssimò a due cifre decimali la preziosa costante. Per tale motivo è anche conosciuto come costante di Archimede (da non confondere con i numeri di Archimede). L'altro studioso a cui è legata la costante è Ludolph, infatti per alcuni è il numero omonimo. Archimede tentò di calcolare pi greco, o meglio la lunghezza di una circonferenza attraverso il calcolo dei perimetri di figure piane inscritte e circoscritte il cerchio con un numero di lati sempre maggiore. Aumentare i lati porta ad una migliore approssimazio-

ne della circonferenza. Altri importanti contributi sono dovuti a Descartes. Egli procedette in una direzione opposta a quelle fino a quel momento adottate che prevedevano la "quadratura di un cerchio". In altre parole gli studiosi fino ad allora tentavano con vari metodi di costruire un quadrato partendo da una circonferenza corrispondente. Descartes con la regola del compasso, partì dal perimetro del quadrato e tentò di calcolare la circonferenza. Arrivò all'importante relazione $p/d=\pi$. Tale metodo è conosciuto come isometrie. Ma non finisce qui!
Il pi greco non ha solo aspetti matematici come molti potrebbero pensare.

RIFERIMENTI SULLA RIVISTA

Articoli correlati su ioProgrammo di Fabio Grimaldi:

n. 35 - Montecarlo risolvere problemi ricorrenti con i numeri casuali

n. 20 - Primo estratto sulla ruota di ...

UN PRIMO METODO PER CALCOLARLO

Partendo da una delle numerose serie aritmetiche presentate, quella di Leibniz, costruiamo un algoritmo che calcoli il valore di pi greco. La serie è costituita da infinite frazioni, è risulta ovvio come il valore che cerchiamo sia più preciso per un numero maggiore di termini. La particolarità è che le frazioni si alternano di segno. Inoltre, dall'analisi dei risultati si evidenzia come tale serie sia alquanto "lenta" nel convergere a valori "di apprezzabile precisione". In C++ il problema si risolve impostando un ciclo di m iterazioni, con m valore di input.

```
// Calcolo di pi greco con il metodo di Leibniz
void leibniz()
{
   unsigned long m,j;
   float q,k,sum,s,x;
   cout<<"n. prove -> ";
   cin>>m;
   sum=1;
   x=0;
   for (j=1; j<=m; j++)
   {
      x++;
      q= 1/(x*2+1);
      // si alternano fattori positivi e negativi
      s=((j % 2) == 0)? +1:-1;
      sum=sum+q*s;
      k=4*sum;
      cout.precision(10);
      cout.setf(ios::left, ios::adjustfield);
      cout<<j<<" -> "<<k<<'\n';
   }
};
```

JIUZHANG SUANSHU

Il migliore e più influente scritto antico cinese, che si presenta in forma organica come nove capitoli che affronta elementi computazionali è lo Jiuzhang Suanshu. Composto durante la dinastia di Han, tra il 220 AC e il 206 DC, l'opera contiene 246 problemi. Tra questi si affronta il calcolo dell'area del cerchio, oltre che di altre figure come piramidi e sfere. Per il calcolo della superficie circolare la procedura proposta su questi antichi scritti considera una costante pari a 3. Si tratta quindi della prima approssimazione del valore di π.

Nella variabile sum si aggiungono ad ogni iterazione del ciclo i nuovi termini calcolati come rapporto tra uno e numeri dispari man mano crescenti di un fattore 2, si tratta della variabile *q*. Con *s* si assegna: il segno negativo per iterazioni dispari; positivo altrimenti. Sintetico è l'*if* aritmetico che calcola *s*. In **Figura 1** è riportato l'output della routine *(k)* sviluppata per soli trenta termini.

```
C:\aa\edmaster\ioprogrammo\n_99\soluz\cod
n. prove -> 6

punti   nhit    nmiss   k
33      28      5       3.393939495
44      41      3       3.727272749
132     110     22      3.333333254
704     558     146     3.170454502
5866    4636    1230    3.161268234
70392   55553   14839   3.156779289
```

Fig. 1: **Output del programma sviluppato per il calcolo di pi greco con il metodo di Descartes**

Ovviamente, per ottenere risultati apprezzabili è necessario iterare un numero elevato di volte, inserire quindi grandi valori di *m*.

IL METODO MONTECARLO

Un sorprendente incontro tra il concetto che sembrerebbe prettamente teorico di pi greco e la programmazione, si ha nel *"metodo montecarlo"*. Supponiamo di trovarci in una giornata nuvolosa, tendente a pioggia, a chiederci di trovare un metodo alternativo per calcolare pi greco. Segniamo su un grande cartone di misura rigorosamente quadrata un cerchio iscritto. Poniamo all'aperto il cartone. Nel frattempo cadranno le prime gocce che colpiranno inevitabilmente il nostro manufatto. Con pazienza segneremo il numero di gocce che cadono all'interno della circonferenza e quelle all'esterno. Chiameremo le prime *hit* e le seconde *miss*. Ad un certo punto rientriamo al calduccio della casa e contiamo il numero di hit *(nhit)* e il numero di miss *(nmiss)*. La natura del tutto casuale con cui cade la pioggia ci induce a pensare che il numero totale di gocce *(nhit + nmiss)* sia proporzionale all'area del quadrato; mentre quelle interne al cerchio *nhit* sia proporzionale all'area della figura circolare. Ne consegue che si riesce ad avere un valore, ovviamente approssimato dell'area della circonferenza, impostando la semplice proporzione.

$$\frac{Ac}{nhit} = \frac{Aq}{nhit+nmiss}$$

Possiamo esprimere l'area del cerchio come il prodotto tra una costante e il quadrato del raggio, *Ac=k*r*r*. L'obiettivo del nostro procedimento è proprio calcolare la costante *k*. Se il ragionamento è giusto *k* sarà proprio pi greco. Sulla base di queste considerazioni e del fatto che il raggio del cerchio è la metà del lato del quadrato, per come abbiamo costruito l'esperimento, allora possiamo isolare la nostra costante e vedere qual è la formula che la descrive.

$$Ac = Aq\,\frac{nhit}{nhit+nmiss}$$

$$Ac = kr^2 \quad l=2r$$

$$k = \frac{(2r)^2}{r^2}\,\frac{nhit}{nhit+nmiss}$$

$$k = 4\,\frac{nhit}{nhit+nmiss}$$

L'ultima formula esprime in modo definitivo *k*, indica che è sufficiente calcolare i due insiemi di punti interni ed esterni al cerchio e avere una stima della costante *k*. Si tratta di un metodo stocastico, poiché fa uso di numeri casuali. Per rendere le cose più precise ed attendibili si genera la pioggia al calduccio di una stanza con un potente elaboratore. La pioggia che sarà quindi elettronica fornirà una stima della costante. Vedremo come il programma proposto di seguito migliora la stima con l'aumentare dei punti, che è ovviamente relazionata alla bontà del generatore di numeri casuali. Per semplificare, nella routine sviluppata si è considerato un quarto di quadrato e quindi un quarto di cerchio, per intenderci il solo quadrante positivo degli assi cartesiani. Poiché la costante è proporzionale al rapporto tra le due aree non si dovrà cambiare nulla rispetto alla formula che abbiamo appena ottenuto. Ecco il codice.

```
// Calcolo di Pi greco
// con il metodo montecarlo
void pgrmc()
{
   const long raggio=999;
   time_t t;
   // l'ora è utilizzata come seme
   // per la generazione di numeri casuali
   unsigned long i,j,n,m;
   float x,y,d,k,nhit,nmiss;
   srand((unsigned ) time(&t));
   n=100;
   cout<<"n. prove -> ";
   cin>>m;
   cout<<"\npunti\tnhit\tnmiss\tk\n";
   for (j=1; j<=m; j++)
   // si fanno più prove con un numero sempre
   // maggiore di punti casuali
   {
     n=j*j*n/3;
     nhit=0;
     nmiss=0;
     for (i=1; i<=n; i++)
     {
   x=rand()%(raggio+1);
        y=rand()%(raggio+1);
        d=sqrt(x*x+y*y);
        if ( d>raggio ) nmiss++;
        else nhit++;
     };
     k=4*nhit/(nhit+nmiss);
     cout<<n<<'\t'<<nhit<<'\t'<<nmiss;
     cout.precision(10);
     cout.setf(ios::left, ios::adjustfield);
     cout<<'\t'<<k<<'\n';
   }
}
```

Fig. 2: ***Output del programma sviluppato per il calcolo di π con il metodo di Montecarlo***

Il doppio ciclo innestato è stato introdotto per potere comparare risultati diversi corrispondenti a numeri diversi di punti casuali. Il range di generazione dei numeri va da 0 a 999, cosicché il raggio è mille. La funzione *srand* inizializza il seme di generazione dei numeri casuali, a cui ho introdotto l'elemento variabile tempo (l'ora). Nel ciclo interno si producono le coordinate nel range descritto e si verifica se siano hit o miss, quindi si contano. Alla fine si calcola *k*. L'output prodotto è riportato in **Figura 2**. In effetti l'approssimazione è soltanto alla prima cifra decimale, certamente risulterebbe migliore per un maggiore numero di punti. Purtroppo alcuni limiti dettati dalle dimensioni delle variabili impediscono analisi più approfondite. A tale scopo sarebbero appropriati ambienti di sviluppo numerico come il Matlab.

ALTRE CURIOSITÀ

Legate a π vi sono altri interessanti curiosità. Ad esempio per molti è una questione di primaria importanza ricordare il maggior numero di cifre decimali. A ta-

IN NATURA

Il prof H.H. Stølum, uno scienziato della terra dell'università di Cambridge, ha riscontrato una singolare analogia. Secondo i suoi studi il rapporto tra la lunghezza di un fiume e la linea d'area dal sorgente alla foce è un numero che in media è maggiore di 3 e tende sorprendentemente proprio a *p*. Nel caso dei fiumi lo scalare esprime in sintesi un rapporto tra ordine e caos, i due elementi diametralmente opposti che da sempre descrivono le linee di studio delle scienze. Nel caso specifico anche il contributo del grande Einstein porta alle stesse conclusioni. Secondo il Nobel, infatti, i fiumi man mano che si avvicinano alla foce tendono a creare maggiori erosioni portando a maggiori curvature.
Gli studi riscontrano che l'approssimazione di tale rapporto è maggiore è più vicina a *p* per quei fiumi che affrontano tenui dislivelli come accade in Brasile o in Siberia.

le proposito sono stati introdotti molti metodi per aiutare la memoria. Il più usato prevede la memorizzazione di sequenze di parole, come filastrocche il cui numero di lettere corrisponde in ordine alle cifre decimali di pi greco. Il più bello in italiano che ho trovato in rete è firmato Pasquale, eccolo:

"Noi e loro, a volte, bisognamo di notare cifre fra molte, affinché calcolare possiam lunghezze, Con il mio versetto quel numero si arreca. Dici che son prodezze? Fai un risetto, letterina greca!".

Le parole noi e loro hanno lunghezza rispettivamente: 3,1 e 4 e così via. Eccone uno inglese molto carino:

"How I want a drink, alcoholic of course, after the heavy chapters involving quantum mechanics. One is, yes, adequate even enough to induce some fun and pleasure for an instant, miserably brief".

Che più o meno significa:

"Sì, ho bisogno di un drink, alcoolico naturalmente, dopo le pesanti lezioni sulla meccanica quantistica..."

Sei un cervellone? Puoi verificarlo al sito di una scuola di Trento: *http://www.itccarli.it/Matematica/carlpigr.php*, qui si controlla se la propria data di nascita è una sequenza numerica contenuta nel primo milione di cifre decimali di pi greco. Se così fosse, secondo una sorta di leggenda si è una mente matematica. Per la cronaca la mia data di nascita è presente dopo la *195402-ma* cifra. Si può fare lo stesso gioco immettendo un nome.

• SEGMENTI
V. Aieta
(Prometeo)

• A HISTORY OF ALGORITHMS
J.L Chabert, Al
(Springer)

• L'ULTIMO TEOREMA DI FERMAT
S. Singh
(BUR Libri)

ANEDDOTO DI JOHN CONWAY

Al sito http://utenti.quipo.it /base5/numeri /pigreco.htm si trova un simpatico aneddoto riguardante il famoso studioso. "Un giorno decisi di imparare a memoria le prime mille cifre del pi greco - ricorda Conway - stimolato da mia moglie Larissa, una matematica di origine russa, che aveva bisogno del valore di p e non ricordava che 3,14. Le insegnai le prime cento cifre che ricordavo già a memoria. Ma questo a lei non bastava e, visto che anch'io non sapevo andare oltre, decidemmo insieme di programmare lo studio di cento nuove cifre ogni giorno, per arrivare almeno a mille, da imparare nei momenti in cui eravamo insieme, al di fuori del nostro lavoro"."È stato divertente - continua Conway - perché ogni domenica facevamo una passeggiata fino a Grantchester, una graziosa, piccola cittadina vicino a Cambridge e lungo il percorso recitavamo a turno i gruppi successivi di 20 cifre del p, come fossero piccole poesie. Venti cifre io e venti cifre mia moglie e così di seguito, alternandoci nella recita: in questo modo siamo arrivati a memorizzare le mille cifre del pi greco".

IL NUMERO AUREO

Il numero in questione è ancora irrazionale. Ecco una sua approssimazione a qualche cifra decimale 1,68103 39887 49894 84820 45868 .. il suo nome è ϕ ma è anche conosciuto come numero aureo. Chi non ha letto *Il codice Da Vinci*, si è trattato di un fenomeno contagioso, direi dai risvolti positivi, che ha colpito un po' tutti. Sicuramente allora ricorderete il numero. La prima sorpresa è che il numero è legato alla sequenza di Fibonacci, per la quale un numero è la somma dei due precedenti. Si tratta del rapporto tra due numeri consecutivi che converge proprio a ϕ. Gli esempi dove compare ϕ come rapporto tra grandezze sono tanti. L'uomo vitruviano, il famoso nudo maschile di Leonardo DaVinci è il simbolo della perfezione e dell'armonia umana.
Bene, i rapporti tra le diverse parti del corpo convergono sempre al numero d'oro. Altezza dell'uomo fratto altezza dell'ombelico, la distanza spalla - punta delle dita fratto gomito - punta delle dita e molte alte frazioni sono proprio ϕ. In geometria viene definita la sezione aurea. Un segmento *a* viene diviso in una sezione aurea se *x* è la parte maggiore e *a-x* la minore (**Figura 3**), e vale la relazione

$$\frac{a}{x} = \frac{x}{a-x}$$

In altri termini quando il quadrato costruito su *x* ha la stessa area del rettangolo (*a-x*), *x*. Il rapporto *x/a* è proprio il numero aureo. Il rettangolo ottenuto è definito aureo. Secondo gli psicologi è il più bello ed armonioso. Anche Pitagora nei suoi pentagrammi riporta come rapporto tra i segmenti ϕ così come Euclide che chiama questo rapporto come "media ed estrema ragione". Se dividiamo un rettangolo aureo in un quadrato, la parte rimanente sarà ancora un rettangolo aureo. Si può iterare il procedimento e creare così infiniti rettangoli aurei. La spirale logaritmica inscritta unendo i vertici dei rettangoli che via via vanno generando è un altro elemento che troviamo ancora una volta in natura. La tela costruita da un ragno segue tali traiettorie, così come alcune corna di animali. Ancora si ritrova nella disposizione dei semi di girasole. In senso orario crescono 55 semi mentre nel senso contrario 34. Il rapporto è ancora *f*.

CONCLUSIONI

Se il numero di link prodotti da un motore di ricerca è un metodo per valutare la popolarità di un argomento. Un altro modo per convalidarne anche la qualità è verificare chi si è occupato nel corso degli anni dell'argomento. In tal senso i nomi emersi nel corso della trattazione toccano praticamente le più brillanti menti dell'umanità da Archimede a Pitagora, Euclide a Eulero e ancora da Riemann a Descartes e sono una garanzia che stiamo percorrendo la strada giusta.

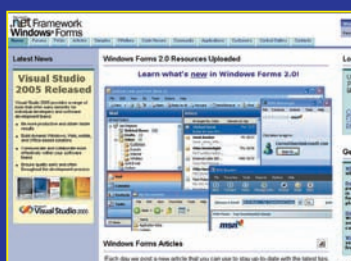
Fabio Grimaldi

ON LINE

SHARPTOOLBOX

Un sito ricchissimo di librerie, addin e molto altro sulla programmazione .NET. Tutto accuratamente recensito e diviso per categorie, per una consultazione semplice.
http://www.sharptoolbox.com

CODE GENERATION

Articoli molto tecnici sull'ingegneria del codice in questo interessante sito, che si distingue proprio per la tipologia di informazione, in un settore che d'altra parte non sempre è considerato quanto dovrebbe.
http://www.codegeneration.net

WINDOWS FORMS

Il sito ufficiale sul framework .NET per quanto riguarda la programmazione Desktop Oriented, frequentato da alcuni dei maggiori "Guru" della programmazione.
http://windowsforms.net

Biblioteca

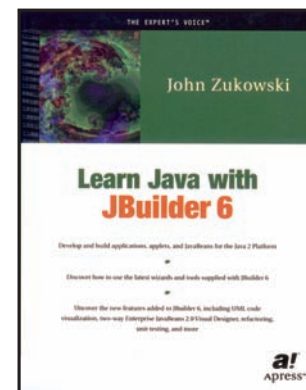
COPYLEFT & OPENCONTENT

In un epoca in cui il movimento OpenSource sta radicalmente cambiando la concezione del software e del diritto d'autore, un libro come questo si pone come un punto di riferimento certo per chiunque voglia almeno tentare di capire come muovere i primi passi nel complesso mondo delle licenze, sia da un punto di vista etico che da un punto di vista legale. Il libro è completamente in Italiano scritto da Simone Aliprandi, questo fa si che le nozioni sul CopyLeft siano completamente calate in un panorama, il nostro, diverso per molti aspetti da quello americano dove il fenomeno ha avuto origine e si mostra con sfaccettature molto più sentite rispetto al nostro paese. All'interno del libro trovano posto i testi delle principali licenze, debitamente commentate e ricche di informazioni a margine. Nel complesso un libro che ben si cala nella realtà dei nostri giorni, utile per agire in modo eticamente corretto ma anche per capire come un modo diverso di concepire il software e in generale il diritto d'autore possa essere anche un modo per rendere remunerativo il proprio lavoro.

Difficoltà: Bassa • **Autore:** Simone Aliprandi • **Editore:** PrimaOra • **ISBN:** 88-901724-0-1 • **Anno di pubblicazione:** 2005 • **Lingua:** Italiana • **Pagine:** 176 • **Prezzo:** € 12,00

LEARN JAVA WITH JBUILDER 6

Di Jbuilder parliamo abbondantemente in questo numero di ioProgrammo, presentandone nel Cd allegato anche la versione Foundation Edition. Jbuilder è un prodotto complesso, quanto potente che riunisce sotto un unico "abito" centinaia di tool che si possono trasformare in opportunità per il programmatore. La vocazione RAD i Jbuilder inoltre ben si presta a diventare un punto di riferimento per coloro che vogliono imparare a programmare in Java. Di fatto l'essere costretti ad utilizzare dei modelli solidi ed affidabili è un buon modo anche per capire qual è uno stile di programmazione adeguato, e per iniziare a muovere i primi passi con il proprio linguaggio preferito. Rappresenta da un lato un una guida all'ambiente, dall'altro una guida di riferimento puntuale e precisa al linguaggio. I temi affrontati sono: dal disegno delle interfacce grafiche tramite l'ambiente RAD di Jbuilder, all'accesso ai file, all'internazionalizzazione e all'uso dei Jbeans.

Difficoltà: Media • **Autore:** John Zukowski • **Editore:** APress • **ISBN:** 1-893115-98-4 • **Anno di pubblicazione:** 2005 • **Lingua:** Inglese • **Pagine:** 480 • **Prezzo:** € 35,00

C++ FONDAMENTI DI PROGRAMMAZIONE

Il C++ rimane uno dei linguaggi principe su cui si basano un numero incredibile di applicazioni. Il fascino esercitato da questo linguaggio su milioni di programmatori d'altra parte è evidente. Il libro dei fratelli Deitel costruisce un percorso affascinante attraverso l'intricato mondo del C++. Si parte dalle basi, quali la sintassi e la filosofia del linguaggio per poi approdare a concetti complessi quali ad esempio il polimorfismo.
La struttura del libro è molto modulare e consente un approccio per esempi, molto graduale che mette in grado il programmatore neofita di avvicinarsi al mondo del C++ con semplicità estrema. Nonostante il target del libro sia quello dei nuovi programmatori, tuttavia ogni concetto è espresso con la massima attenzione ai dettagli, di modo che anche chi è già un esperto trova comunque sempre qualche dettaglio da apprendere, per un linguaggio come il C++ che d'altra parte non si finisce mai di imparare.

Difficoltà: Media • **Autore:** H.M. Deitel, P.J. Deitel • **Editore:** Apogeo • **ISBN:** 88-503-2386-7 • **Anno di pubblicazione:** 2003 • **Lingua:** italiana • **Pagine:** 742 • **Prezzo:** € 45,00